AD-A059 799 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO F/G 9/2
ACQUISITION OF EMBEDDED COMPUTER SOFTWARE. A DESCRIPTIVE MODEL.(U)
1977 J K WATSON

UNCLASSIFIED AFIT-CI-78-121T NL

1 OF 1
AD A059799

END
DATE FILMED
12-78
DDC

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

DNW

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: CI 78-121T	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): Acquisition of Embedded Computer Software: A Descriptive Model	5. TYPE OF REPORT & PERIOD COVERED: Thesis
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Jerry Keith Watson	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS: AFIT student at the University of Missouri, Rolla, Missouri	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: AFIT/CI, WPAFB OH 45433	12. REPORT DATE: 1978
	13. NUMBER OF PAGES: 72 Pages
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

LEVEL II

16. DISTRIBUTION STATEMENT (of this Report)

Approved for Peblic Release, Distribution Unlimited

DDC RECEIVED OCT 4 1978 F

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

SEP 22 1978

JOSEPH P. HIPPS, Major, USAF
Director of Information, AFIT

APPROVED FOR PUBLIC RELEASE AFR 190-17.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

(6) ACQUISITION OF EMBEDDED COMPUTER SOFTWARE: A DESCRIPTIVE MODEL.

BY

(10) JERRY KEITH WATSON, 1943-

A THESIS

(9) Master's thesis,

Presented to the Faculty of the Graduate School of the

(12) 79 p.

UNIVERSITY OF MISSOURI-ROLLA

In Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE IN ENGINEERING MANAGEMENT

(14) AFIT-CI-78-121T (11) 1977

Approved by

Alfred Omenzetter (Advisor) R L Carmichael

James T. Downing

012 200

ABSTRACT

The purpose of this study is to investigate the process of acquiring Embedded Computer Systems (ECS) within the United States Air Force and the Department of Defense. The major objective is to provide a descriptive model of the Embedded Computer System acquisition process. It is recognized that methods, procedures, rules and regulations applying to ECS acquisitions are dynamic in nature. This model therefore depicts the ECS acquisition process in its current state. This study is intended to be of assistance to those charged with managing acquisitions and developing solutions to the many problems of acquiring ECS. Definitions of terms and concepts which become troublesome in ECS procurements are provided and software is placed in proper perspective with hardware. The entire acquisition process for Embedded Computer Systems is described to serve as a source document for system managers. Another objective is to remind the seasoned software manager that concurrent with elevating the status of software to a major area of concern in ECS, we must not lose sight of the importance of hardware.

The primary emphasis of the model is on the interrelationships between the hardware and software tasks to be accomplished during full scale development. It is intended to cover the acquisition process from a conceptual point of view, rather than provide a "cookbook" approach to acquiring Embedded Computer Systems. The focus is to introduce the concept of an interdependent system of hardware and software development for ECS. Even though many concepts are the same for both Automatic Data Processing (ADP) and Embedded Computer Systems, no attempt was made to cover ADP systems.

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank my thesis committee Dr. Yildirim Omurtag, Dr. Ronald Carmichael and Prof. Vernon Loesing, for the assistance and encouragement that they have given me during the past year. I wish to extend my appreciation to Dr. Omurtag for the many hours spent in technical discussions which led to clarification and refinement of the concepts presented in this thesis. I would especially like to thank my wife, Vicki, for her encouragement throughout the ordeal.

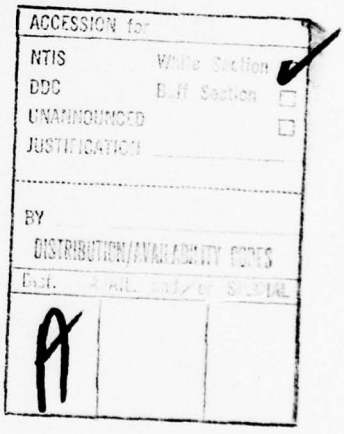


TABLE OF CONTENTS

Table of Contents (continued) Page

LIST OF ILLUSTRATIONS

I. INTRODUCTION

Today, almost all major Department of Defense (DOD) weapon systems are dependent to some extent upon internal computer systems. Large systems such as the B-1 Bomber, Trident Submarine or the F-15 Air Superiority Fighter are critically dependent on the proper operation of these Embedded Computer Systems. A key component of these electrical-mechanical systems has become the computer and its integrated software.

A. DEPARTMENT OF DEFENSE SOFTWARE COSTS

One of the most perplexing and urgent problems now facing the DOD is how to control the mushrooming costs of the software needed to operate Embedded Computer Systems. The DOD now spends billions of dollars annually on software. These costs are projected to continue to rise in the foreseeable future. In some recent major programs the cost of embedded computer software has been three times the cost of its accompanying hardware. (1) Intuition might lead us to believe that, if software is three times as expensive as hardware, managers are paying three times as much attention to software as hardware. Sadly, this has not been the case. Only in the last few years has higher management within the DOD focused on the problem of managing the embedded software acquisition process. The situation has become so serious that almost the entire issue of the Defense Management Journal for October 1975 was devoted to the subject of managing software acquisitions.

To illustrate how drastically software costs have risen, we need only look at the cost of software as a percentage of the total costs for computer systems. In 1955 the cost of software was approximately 17% of

the cost for the average computer system, and by 1974 the cost of software had risen to 65-70% of the total computer system costs. The cost of software is projected to rise even further, up to about 85% by the 1980s (see Figure 1). (1) We might ask, just how expensive is embedded computer software? It has been estimated that DOD annually spends from 3 to 3.5 billion dollars on the various forms of software. Approximately 55% to 75% of these dollars can be classified as weapons systems (embedded) software costs. (1) We can see that an annual savings of only 1% in the cost of software could easily exceed 26 million dollars.

It seems that the spiraling use of embedded computers in major weapons systems has not been accompanied by appropriate management techniques, tailored to the unique environment of acquiring embedded computer software. This lack of management response within both DOD and industry may be one reason for the escalating cost of software. Unlike hardware, software is not something that you can see, feel or touch. It is an intangible product. Consequently, managers and engineers have perhaps concentrated on the things that they could see and visualize easily (hardware), and neglected the software allowing it to become a major source of problems in the total system acquisitions process.

B. THESIS OBJECTIVE

The purpose of this thesis is to investigate the process of acquiring Embedded Computer Systems (ECS) within the United States Air Force and the Department of Defense. The major objective is to provide a descriptive model of the Embedded Computer System acquisition process.

The first section of the thesis introduces the subject of Embedded Computer Systems and their increasing significance in terms of a system's cost. After a brief review of the complex array of problems that

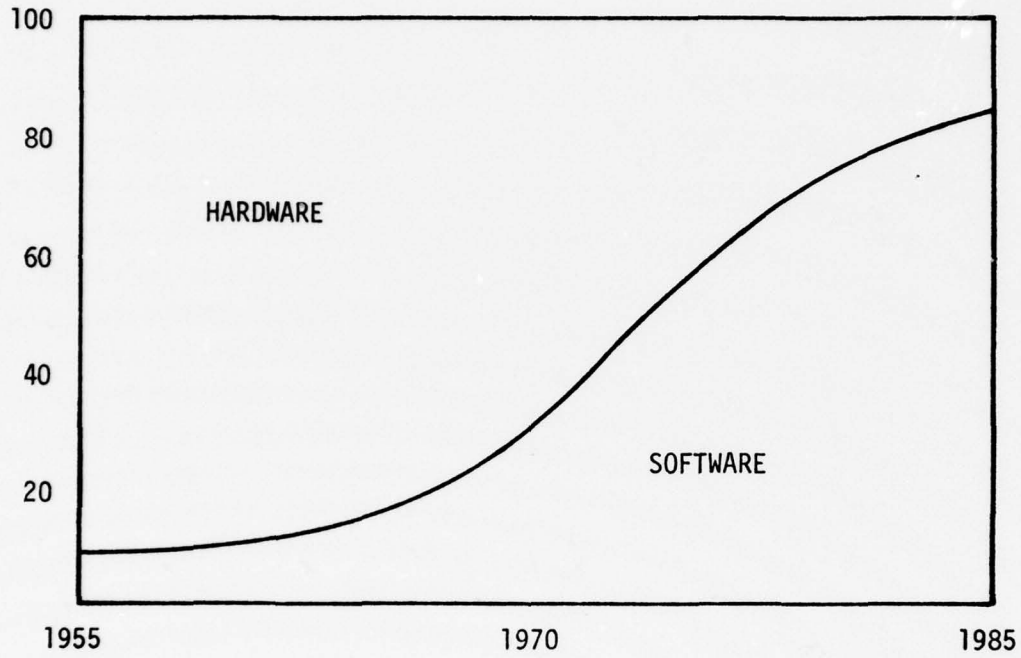


Figure 1. Hardware/Software Cost Trends

confront today's managers, specific concepts and definitions which become troublesome to ECS procurement managers are developed. The next section reviews the current literature pertaining to ECS. Then the ECS acquisition life cycle model is developed consisting of concept formulation, validation, full scale development, production, deployment, operation, maintenance and retirement.

The primary emphasis is on the interrelationships between the hardware and software tasks to be accomplished during full scale development. It is intended to cover the acquisition process from a conceptual point of view, rather than provide a "cookbook" approach to acquiring Embedded Computer Systems. The focus is to introduce the concept of an interdependent system of hardware and software development for ECS. Even though many concepts are the same for both Automatic Data Processing (ADP) and Embedded Computer Systems (ECS), no attempt was made to cover ADP systems. The study is intended primarily for the manager who is new to the acquisition of Embedded Computer Systems.

II. LITERATURE REVIEW

Management disciplines have evolved and kept pace with the innovations and changing hardware technologies. This has allowed great increases in computer hardware capability, while at the same time bringing about reductions or at least moderate increases in the costs. Software management disciplines, however, have not kept pace with the demands and expectations attendant with increased dependency of weapon systems on embedded computers. Concern within the DOD about ECS management is evidenced by the rapidly increasing number of government reports and studies on the subject. A quick review of the Government Reports Index or Defense Information Exchange Service listing on software management will show a yearly escalation in the numbers of articles since 1971.

In reading the software literature of the last three years, there is a wealth of work being done on the individual problems identified in Figure 2. (2) Articles discussing particular programs, programming techniques or reliability of software are numerous. Singularly lacking are articles attempting to put the entire system of embedded computer software acquisition in perspective. Boehm's excellent 1973 article assessing the impact of software is an admirable job of putting software in perspective and setting the stage for further data gathering and research.

Our efforts will begin by a review of the problems and continue with existing solutions before developing a descriptive model. A critical examination of the model from the current users' points of view will follow.

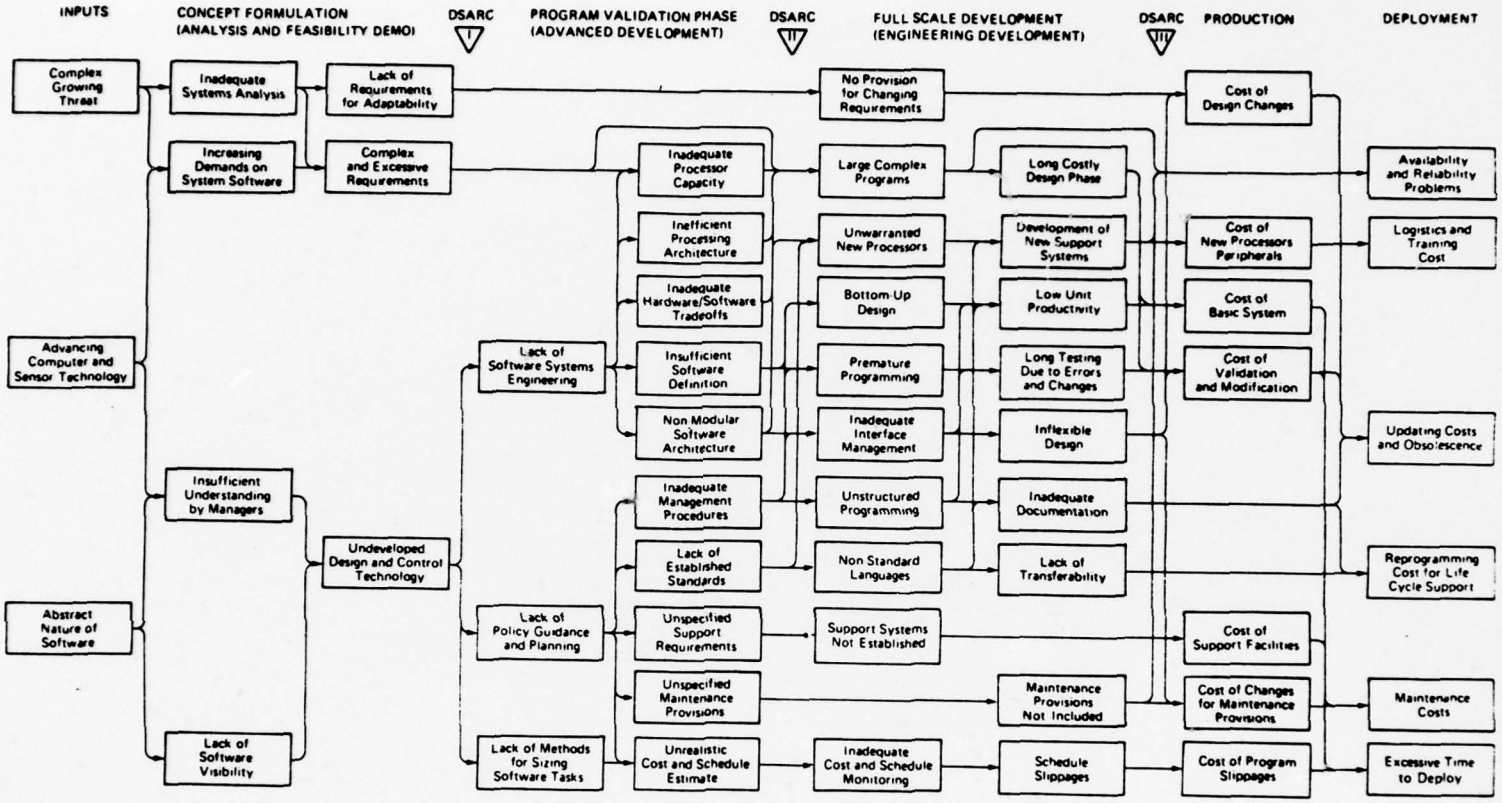


Figure 2. Interrelation of Software Acquisition Study Findings

A. REVIEW OF PROBLEMS

No discussion of the procurement process for software would be complete without some mention of the specific problems associated with acquisition. In a recent DOD study 55 general categories of problems were listed. (2) Each phase of the acquisition cycle was depicted with its own distinct problems. Each problem shown in Figure 2 has a causal relationship, and a cascading effect on the succeeding problems. The interrelationship of these problems is the subject of a major portion of the current research in this field. An examination of these problems leads us to the conclusions which were reached by Deputy Assistant Secretary of Defense Jacques S. Gansler, who stated that the most critical problems now facing the DOD and industry in ECS are: (3)

1. Insufficient control over rapidly growing software expenditures.
2. Insufficient research and development in software production.
3. The need for improvements in weapons systems software management.

Recent studies and articles written about the problems inherent in developing computer systems lament as a major problem the lack of visibility into the abstract nature of the process of converting systems (mission) requirements into a viable system of software and hardware. After studying the results of 10 major DOD sponsored studies concerned with the procurement of Embedded Computer Systems, The Johns Hopkins University Applied Physics Laboratory report stated that:

> (1) "The poor understanding of software is generally agreed to contribute to the poor management of Embedded Computer Systems procurements."

and that,

> (2) "A wide variation exists in the degree to which program managers (SIC) are staffed with personnel competent in systems engineering and software applications." (2)

We find by reading the current literature that consideration of software as a subcomponent of major importance (equal or exceeding hardware) has been only a recent development. This increased elevation of the status of software is the direct result of the shifting of the majority of expense from hardware to the software over the last three decades (see Figure 1).

This theme, the lack of understanding of the basic underlying relationships (between hardware and software), was further emphasized by Davis in his paper presented to the Joint Logistics Commanders Software Reliability Work Group. Davis said that:

> (1) "Much of the approval chain in computer systems (both procurement and R&D) is made of people who are not up to speed in contemporary computer business, and not sufficiently supported by people who are." (4, 39)
>
> (2) "There is an insufficient number of skilled software workers in research, technique development and practice." (4, 40)
>
> (3) "Computer systems are often considered as hardware apart from software. This failure to consider the total system (both hardware and software together) has caused many problems." (4, 41)

Wolverton and Schick confirmed that the underlying premise for improving reliability and solving problems of acquiring software ". . . requires understanding of the total software development and test cycle." (5)

The lack of understanding (by managers) was a major theme of the summary report of the Joint Logistics Commanders Electronic Systems Reliability Work Group (6) and the studies performed by both the Johns Hopkins Applied Physics Laboratory (2) and the Mitre Corporation. (7) The findings of all three groups were that:

> "DOD and the services have compiled a large number of regulations, directives and standards for systems acquisition management. In general these directives were written for hardware and do not focus on software issues. DOD standards have almost exclusively revolved around one aspect of software development and acquisition: software configuration management. Attempts have been made to borrow terminology and corresponding regulations and standards from the hardware world, unsuitably modify and apply them to software ... As a result, the number of documents has proliferated and a number of inconsistencies and conflicts between them exist." (6)

In 1974 the Assistant Secretary of Defense (Installations and Logistics) and the Joint Logistics Commanders of all the services established a joint office of the Secretary/Services Weapon System Steering Committee to attack the problems of Embedded Computer Software resource acquisitions. The steering committee issued a proposed capstone directive which was a statement of policies and proposed principles for future directives on software management policies. (8) This directive was the first of what is to be a massive overhaul of all pertinent directives, regulations and military standards dealing with the acquisition of Embedded Computer Systems. This overhaul is designed to correct inconsistencies in the coverage of current regulations and directives, and to develop a consistent coordinated systems methodology for dealing with the acquisition of Embedded Computer Systems. Actions are now underway to:

> "Prepare and maintain appropriate guidance documents (e.g. guidelines, checklists, handbooks and descriptive examples) covering requirements definition, development, acquisition, operation and support issues attendant to computer software in defense systems." (9)

Mitre Corporation in their study proposed that a series of guidebooks be prepared to guide both the software practitioner and program project managers. (7) Some of these guidebooks are now starting to

appear in DOD channels. (10, 11, 12) The proposed list of guidebooks is as follows:

1. Project Guide to Content Requirement and Audience Needs
2. Regulations, Specifications and Standards
3. Contracting for Software Acquisition
4. Measuring and Reporting Software Status
5. Statement of Work (SOW) Preparation
6. Review and Audits
7. Configuration Management
8. Requirements Specification
9. Software Documentation Requirements
10. Verification
11. Validation and Certification
12. Management Reporting by Software Director
13. Computer Program Maintenance
14. Software Quality Assurance
15. Software Cost Estimating and Measuring

B. CURRENT APPROACHES

Many authors have added much to our understanding of the process of developing and acquiring computer software. Singularly lacking in the current literature is a description of the process for developing and acquiring an integrated system of hardware/software. In reviewing over 600 abstracts and a detailed review of over 100 articles and papers no description was found that could truly be termed an Embedded Computer System acquisition model that described the process of acquiring software and hardware as an integrated system. The current literature

on computer software and ECS can be put into two general categories. The first, which we have already discussed is the category of authors who are investigating and describing their approach to some technical problem. A comprehensive list of problem research areas can be compiled from Figure 2. The bulk of the remaining literature deals with various approaches to the management of computer software acquisition. A partial list of articles dealing with the acquisition of software can be found in the paragraphs below and in the Bibliography.

A major common point found in the second category of literature is the description of the software acquisition process in isolation. The tasks and events which occur during software acquisition are related with no attempt to describe the hardware tasks and events which should be occurring simultaneously. One of many examples of this isolated description of the software acquisition process can be obtained from the paper written by Zabriskie. In his paper, "Development of Weapon Systems Computer Programs: Guidelines for Controlling During FSD," (13) Zabriskie does an admirable job of clearly detailing a process for the development of computer software. This article provides a comprehensive list of tasks to be performed during thirteen stages of the software development process. He goes on to provide guidelines during each stage of development and proposes that a separate work break down structure be created for computer programs. Although the paper was very informative, and we should consider it a basic source document, Zabriskie covers only software development activities and does not relate hardware tasks.

With one or two notable exceptions, little attempt has been made to relate the software development and acquisition activities to the DOD weapons systems acquisition phases of concept formulation, validation,

full scale development, production, operation and maintenance. Etheredge presents a general model of the software development activities for automatic data processing systems. (14) This is a three-step model consisting of: 1) analysis and design, 2) implementation and test, and 3) delivery and maintenance. Like Zabriskie, Etheredge discusses acquisition activities only during the full scale development phase and does not relate these activities to hardware activities. One notable exception is the article written by Nelson. In his "Management Handbook for the Estimation of Computer Programming Costs" (15) Nelson describes a six-phase acquisition cycle for automatic data processing software. Each of these phases has been related to the Air Force acquisition cycle as described by the 375 series of manuals. No attempt, however, is made to relate these activities to the activities necessary to procure ECS, nor is hardware discussed.

The literature separates the acquisition process for software into from three to thirteen separate distinct activities or phases. A partial list of the number of phases proposed by the various authors is as follows:

1. Etheredge--three phases (14)
2. Merwin--four phases (16)
3. Capps--five phases (17)
4. Nelson--six phases (15)
5. Mathis and Willmorth--nine phases (18)
6. Bucciarelli--eleven phases (19)
7. Zabriskie--thirteen phases (13)

These proposed models of the software phases during acquisition, all basically contain the same activities. The models differ in their

groupings and the titles assigned to each phase. The confusion created by the many varied models will not be cleared up until the acquisition activities are directly related to the concept formulation, validation, full scale development, production and operation/maintenance phases of the DOD acquisition model.

To summarize, we can say that the current literature contains a number of common approaches. A large portion of the literature discusses the acquisition of computer software from an automatic data processing view rather than an ECS viewpoint. This is understandable because the concept of an Embedded Computer System of hardware and software is relatively new. Those who study ECS end up by discussing computer software in isolation from hardware. Little attempt has been made to date to relate ECS to the total DOD acquisition cycle.

C. CONCLUSIONS

For those most intimately involved, the management system for procuring embedded software may seem obvious. But is it really? The answers to the following questions will provide a useful starting point for any research that attempts to solve software problems. What is the normal model of an embedded software acquisition? Is it different from a hardware or strictly software procurement? If different, why is it different? If identical to hardware and automatic data processing procurements, should it be different? Should events happen in a certain sequence? What are the interrelationships between events and tasks? Software is normally on the critical path for an integrated Embedded Computer Systems procurement. Is this because the development of computer hardware precedes the basic decisions and development of

software? Boehm discusses the concept of developing the software first and then developing the hardware. (2) Is this forbidden by the tenets of the acquisition model? If software was developed first, would significant cost savings result?

Only by developing a clear and concise understanding of the acquisition process for Embedded Computer Systems, will we be started on the road to success through better management. Scientists and engineers thoroughly study a new virus, before attempting to develop a vaccine to effect a cure or prevent disease. We must get back to the basic phenomena that we are dealing with and completely understand that before attempting to mold that phenomena to our needs. The next chapter will present a descriptive model of ECS acquisition process towards this end.

III. THE EMBEDDED COMPUTER SYSTEM MODEL

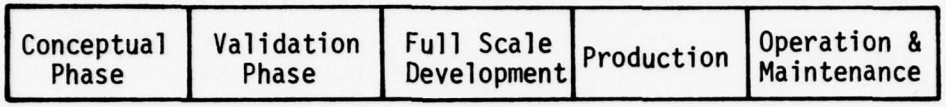
A. TRADITIONAL LIFE CYCLE PHASES

Department of Defense Weapons systems are generally described as going through five phases during their life cycle. The phases in sequence are: concept formulation, program validation, full scale development, production and deployment. A standard graphical description of these phases is shown in Figure 3. Most authors who are primarily concerned with the acquisition process have limited their definitions and concerns to those activities necessary to develop and deploy a new system. Although this study is also primarily concerned with the acquisition process, an expanded depiction is presented so that the reader may have a more accurate picture of a system's life cycle.

Before a new system can be seriously considered for acquisition some change in the environment must occur. The military threat may change or technological advances may be made that allow significant increases in military capabilities. Existing systems may be aging and deteriorating. Many different changes may occur in the environment, but the key is that there must be a felt need before a new system can be conceptualized.

During the Concept Formulation Phase the emphasis is on whether or not the felt need can or should be established as a firm requirement. Various studies are performed to determine if the envisioned systems are technically/economically feasible and if it can be produced in time to realize the desired benefits. Advanced exploratory development is sometimes accomplished during this phase to determine the feasibility of a

TRADITIONAL SYSTEM LIFE CYCLE SEQUENCE



EXPANDED SYSTEM LIFE CYCLE SEQUENCE

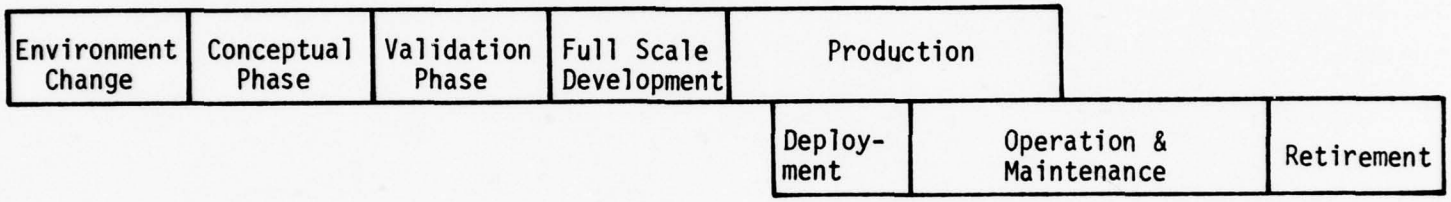


Figure 3. System Life Cycle Comparison

"state of the art" technology, prior to the start of the Validation Phase.

The Validation Phase was formerly called the contract definition phase or project definition phase. During this phase only the minimum preliminary design and engineering necessary to define the system's performance is accomplished. Any design work accomplished during the concept formulation phase is verified. The major technical approaches are validated through extensive analysis and possibly even some hardware development. The last part of this phase results in a contract definition to be implemented during the Full Scale Development (FSD) Phase.

The activities of the Full Scale Development Phase normally include design, prototyping and testing of the new system. The development phase is very complex and will be covered in more detail later.

The Production Phase may cover the production of one item or of many items. In the case of one item, the Production Phase is a continuation of the development process. In production of many items we have the traditional mass production line. The topics of acceptance and quality assurance become increasingly important.

In the Deployment Phase the first production system is taken to an operating location. The activities of this phase can be considered a shake-down cruise prior to placing the system or systems into operation. This phase normally terminates with a formal statement that the system is operational (Initial Operating Capability).

The Operation and Maintenance Phase for major systems usually runs from 10 to 20 years in length or even longer. During this phase the system is operated and maintained, and modifications are made to update capabilities. When the system can no longer fulfill its function

(mission) either through age (degradation) or not being able to keep up with changing threat, it will be considered for replacement and retirement.

The final stage of a system's life is retirement from active use either through permanent retirement or sale.

B. BASIC CONCEPTS

Before continuing the discussion of the acquisition process for software, some clarification and definition of terms is in order. A complete list of software related definitions is beyond the scope of this paper, but a clear understanding of Embedded Computer Systems and embedded computer software is necessary to our investigation.

The first concept that we must understand is software. What is software? We can't see it, feel it or touch it! Software can be many things, depending on your point of view. In some applications software is simply paper. In other applications software is considered as any pliable, soft, flexible product such as rubber or vinyl. For example, the rubber molding around the windshield of an automobile might be considered software. In computer systems the definition of software is not precise. Different authors use different definitions depending on their purpose.

We will use common aspects of various definitions which seem to fit our purposes here. We will consider the term software to be synonymous with the term computer program. So software can be said to be all the computer programs which cause the computer to function correctly, (2) and which are necessary for maintenance, test and modification.

The definition of software normally excludes the accompanying technical documentation. Computer listings, printouts, flow diagrams and

other information which explain the software, are considered descriptions of the software but not the software itself. From an overall procurement and cost viewpoint, we should consider the technical documentation to be not separable from the software. A large portion of embedded software costs are the costs of procuring the technical data to allow understanding, updates and later modification of computer programs by the user as the need arises.

We have previously defined software as being synonymous with computer programs. Now the question becomes, what is a computer program? A computer program or programs can be defined as the series of instructions which are designed to cause that computer to do computations or execute certain functions. (2)

Now that we have a working definition of software, we need to extend our understanding to the concept of Embedded Computer Systems. An Embedded Computer System as used within the DOD is a system which is dedicated to a specific function within a larger system whose primary function is not data processing. Electrical-mechanical systems such as DOD weapon systems with integrated Embedded Computer Systems are enumerable. There are two key characteristics of Embedded Computer Systems, which distinguish them quite well from other computer systems. In an Embedded Computer System the hardware and software are designed and developed simultaneously. In non-embedded (general purpose) computer systems the hardware is developed to be compatible with a multitude of programming purposes. The software is developed separately to work with some general purpose machine and perform some specific function. The embedded computer and its software are developed as an integral part of the much larger electrical-mechanical system. Another characteristic of

note is that the embedded systems are mobile. That is, they are within a moving system with all the implications of miniaturization, cooling and power requirements. (2)

The definitions above are presented to establish the basic software concepts necessary to understanding Embedded Computer Systems. The definitions below are presented to add clarity to the embedded computer acquisition model, which will be presented later.

When a manager or a non-technical person first comes into contact with Embedded Computer Systems, some concepts related to the procurement process are very hard to grasp. Trouble can be expected to arise when a person familiar with a term in an equipment sense must transfer to a computer sense which is subtly different. If the initial understanding is faulty, misconceptions can be carried for years, causing problems or controversy.

This section of the report is by no means intended to convey a complete set of definitions, for either the procurement process or Embedded Computer Systems. A complete and comprehensive set of definitions is available from various military standards and regulations.

Reliability is a term which has a significant change in meaning and the implementation of requirements when switching from the hardware (equipment) to the software (computer programs) sense. Reliability can be generally taken to mean the degree to which satisfactory performance can be expected. The Defense Standardization Manual 4120.3 - M states that,

> "Reliability is the probability of performance (of a given piece of equipment or part) of a specified function without failure under given conditions for a specified period of time." (20, 5-22)

Equipment (hardware) reliability requirements are normally specified as failure rates or the mean time between failures. While this definition may be satisfactory for hardware, it is not entirely satisfactory when we refer to software. First, software does not normally fail, at least in the sense that it suddenly stops functioning. Software does not physically degrade, thus it is not subject to sudden catastrophic failure. Software will almost always repeatedly do what the programmer coded it to do, and yet still may not meet the functional (operation) performance requirements. To give software reliability a meaningful definition, The Joint Logistics Commanders Software Reliability Work Group has defined software reliability as,

> "...the probability that software will satisfy stated operational requirements for a specified time internal or a unit time application in the operational environment." (6, 87)

This allows a definition of reliable software even though it may have errors (programmer, coding or design), yet still performs its operational functions satisfactorily. The Reliability Work Group goes on to say that, "There is no quantifiable means at present to measure software reliability." (6, 89) But they do list some qualitative indicators.

So we can see that although the manager has a definition of reliability for software, it is impossible to prove that the software is reliable. The most that a manager can expect is to gain a feeling for the probability of reliability through historical records of the developmental and operational testing performed.

For many years the DOD has treated software for embedded computers as technical data. This procedure has given rise to some particular problems with the terms validation and verification. It is interesting to note that the term validation appears during the procurement process

in three distinctly different senses. Before a contract is let for developments the project must go through what is termed The Validation Phase. During this effort the preliminary design and engineering concepts for the Embedded Computer System are verified or accomplished and firm contract management planning is performed.

The concept of validation again appears during the development process just prior to the start of production or just prior to operation for a one-of-a-kind system. Validation is used to prove that the system complies with its specified performance.

Air Force regulation 800-14 defines validation of computer programs as the process of determining that the computer programs (software) were developed in accordance with its stated specification, and verification as the process of determining that the computer programs (software) satisfactorily perform in the mission environment, the functions for which it was designed. (21) Notice the word satisfactorily used in the software definition. Because of the impossibility of checking all of the logical paths, the most we can strive for is satisfactory performance coupled with an assurance (through documentation) that the software was developed properly. I repeat we can never prove that this software is 100% reliable. In hardware testing we can physically check all its circuits for proper operation, and except for future degradation, we can say that we have proven its performance.

The terms validation and verification again appear toward the later parts of the Development Phase, when the technical data (repair manuals, users manuals, etc.) are prepared as Technical Orders (T.O.s). In the T.O. sense, validation is the process in which the contractor proves the adequacy and accuracy of the information contained in the document.

Through check-out tests, calibration and other procedures the contractor assures compatibility between the document and the requirements specification for the system. Verification in the T.O. sense is the process by which Air Force personnel test and prove that the T.O. is clear, logical and sufficient for operation and maintaining the system. Verification tests are also used for certifying that the T.O.s are compatible with pertinent hardware, tools and support equipment. This is normally accomplished during the Air Force test phase prior to deployment of the system. (22) A summary of the terms validation/verification might read as follows:

1. In the Validation Phase of a procurement cycle we assure that the preliminary design will meet the needs of the mission, and do advance planning for a later development contract.
2. Hardware validation is the process where the contractor proves that the equipment complies with its specification performance requirements.
3. Software validation/verification is the process where the contractor shows that the software was developed in accordance with the specification and satisfactorily performs.
4. Technical Order Validation. The contractor proves the adequacy and accuracy of the technical document.
5. Technical Order Verification. The Air Force verifies that the document is clear, logical and compatible with the support tools and equipment.

C. MODEL DEVELOPMENT

1. Concept Formulation. The basic purpose of the Concept Formulation Phase is to determine whether or not a proposed system warrants the

further expenditure of funds. Concept Formulation can be considered both a defining and a weeding out process. The major output of this phase is a description of a system which will fulfill the operational needs of the user. The weeding out aspect of the process is the rejection of alternatives that do not satisfy user needs or are sub-optional in terms of cost, schedule or performance. Another definition for the Concept Formulation Phase might read as follows: To provide all the necessary facts, data, analysis, studies and other pertinent information which allows higher level decision makers to determine if a project is necessary. A key attribute of this phase is the minimal expenditure of funds.

The major elements of the Concept Formulation Phase are listed below: (15)

INPUTS:

1. Users' Requirements and Operating Environment
2. Planning Criteria
3. Cost Estimating Techniques and Relationships
4. Resources required/available

FUNCTIONS:

1. Initial System Definition
2. Technological Alternatives Characteristics and Requirements
3. Feasibility Studies, Technical/Economical and Schedule
4. Cost/Benefit Comparisons
5. Selection of Most Promising Alternative
6. Engineering Refinement
7. Draft Functional Specification
8. Advanced Planning

OUTPUTS:

1. Program Description
2. Draft System Performance Specification
3. Preliminary Resources Requirements
4. Tentative Schedules
5. Cost Justifications
6. Subsystem Operating Concepts

Presented below are three figures which depict the Concept Formulation Phase. Figure 4 was taken from AFSCM/AFLCM 375-7. (23) Figure 5 was taken from a Rand Corporation study on the management of software acquisition. (24) Both of these depictions are accurate in their substance.

Looking at these models the reader can readily see that the AFSCM/AFLCM 375-7 model emphasizes the procurement jargon and resulting documents at the expense of clarity. We can also see that the Rand version is so simplified that the reader does not really get a feeling for the weeding out process. A proposed model is presented in Figure 6 to add clarity to the picture of the total concept formulation environment. As shown in Figure 6, a system (project or program) must pass a series of hurdles prior to completing the concept formulation process. These hurdles are designed to optimize the alternative approaches to the problem solution.

It is important to note here that the primary emphasis of this phase is on the major weapons system (aircraft, missile, ship, etc.). The subsystems (such as embedded computers) are considered from a functional viewpoint; i.e., is the performance desired from the subsystem technically possible? Attention to how the performance will be achieved

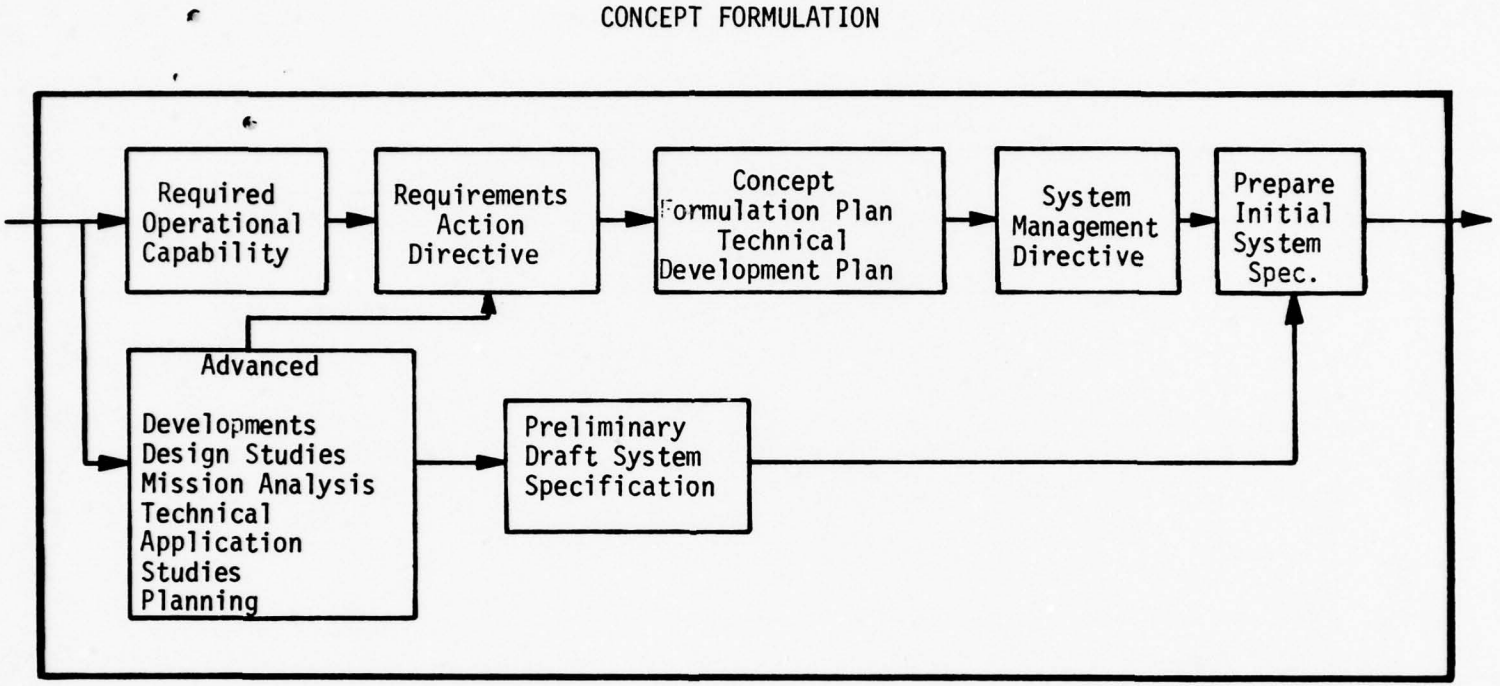


Figure 4. Depiction of Concept Formulation, Air Force Systems Command Manual 375-7

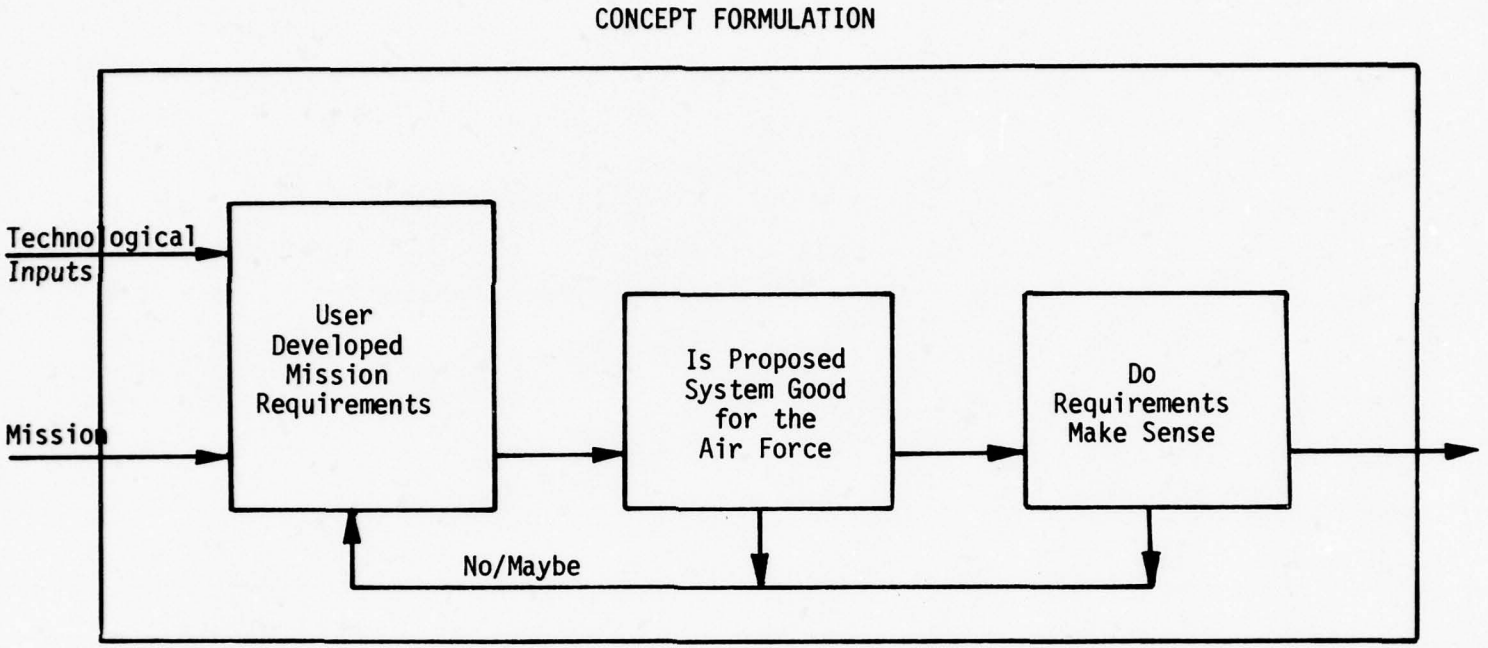


Figure 5. Rand Corporation Depiction of Concept Formulation

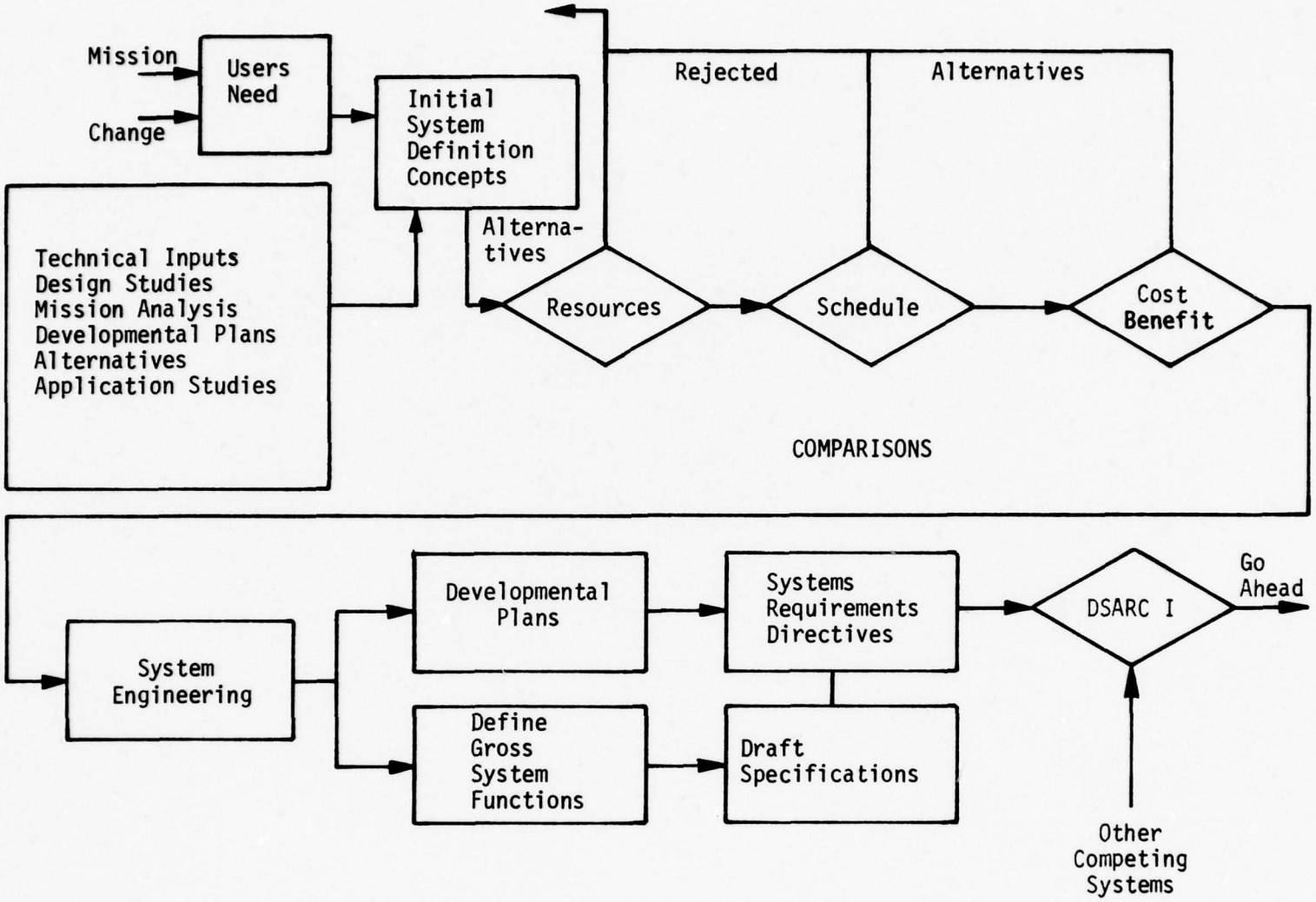


Figure 6. Detailed Concept Formulation Activities

is left to the validation phase. Only if the major system passes the hurdles of concept formulation is it appropriate to expend funds to determine how to achieve the subsystems' performance.

2. Validation. During this phase of activities the systems performance requirements are translated into subsystems performance requirements. These subsystem requirements, all taken together are intended to meet the performance specified for the major system. The first action of this phase is an analysis and evaluation to determine the technical and economic adequacy of the proposed system requirements. The completeness, effectiveness, and any deficiencies in the proposed system are evaluated in light of the users' mission. The degree of technical risk is established. This analysis and evaluation process results in a refinement of the original performance concepts, systems definition and performance requirements.

After the major system has been treated, each subsystem (e.g. avionics, airframe, propulsion, etc.) is analyzed, evaluated and defined in detail. The results of this subsystem activity is an operating concept, performance requirements and a set of design requirements. These concepts and requirements are then compiled into specification documents which will govern all of the following development activities.

In parallel with the technical refining activities are the detailed planning, cost estimating and scheduling activities. Plans are made for the design, development, test, integration and eventual acceptance of each major subsystem as well as the overall system. Detailed cost estimates are accomplished and preliminary schedules are established. The work of this phase is usually a joint effort between the Air Force and a civilian contractor (or contractors), at least for major weapons

systems. The majority of this work is accomplished by the contractor with the Air Force reviewing and approving.

The major elements of the validation phase are listed below: (15)

INPUTS:

1. Mission Requirements
2. User Operating Environment
3. Planning Criteria
4. Subsystem Operating Concepts
5. Cost Estimating Technique and Relationships
6. Program Description
7. Draft System Performance Specification
8. Preliminary Resource Requirements
9. Tentative Schedules
10. Preliminary Cost Estimates

FUNCTIONS:

1. Requirement Analysis
2. Evaluation and Compatibility Studies
3. Developmental Planning
4. Contract Planning
5. Detailed Cost Estimating
6. Trade Off Studies
7. Developmental Tools Planning
8. Interface Definitions
9. Specification Preparation

OUTPUTS:

1. System Specification
2. Subsystem Design Specification
3. Schedules
4. Preliminary Test and Integration Plans
5. Results of Trade Off Studies
6. Detailed Cost/Benefit Studies of Alternatives
7. Requirements Analysis
8. Operational and Developmental Plans
9. Detailed Cost Estimates

A simplified description of activities are contained in both the Rand and AFSCM/AFLCM 375-7 versions of the validation phase, Figures 7 and 8. Both of these figures show the preparation of specifications as the major validation activity. Actually, the validation activities as previously stated are much more comprehensive than just specification writing. Figure 9 shows the major elements of the validation process and their interrelationships. Also shown is the decision point where other proposed DOD systems compete for scarce resources (DSARC II).

DSARC is an acronym for the Defense Systems Acquisition Review Council. This body decides at specific points whether a system should be recommended for funding and continuation into the next phase of its life cycle.

Most authors who explain the acquisition of software (including embedded computer software) do two things. First, they ignore the hardware or only make passing references; second, they start the Concept Formulation Phase with the analysis of software requirements and carry this

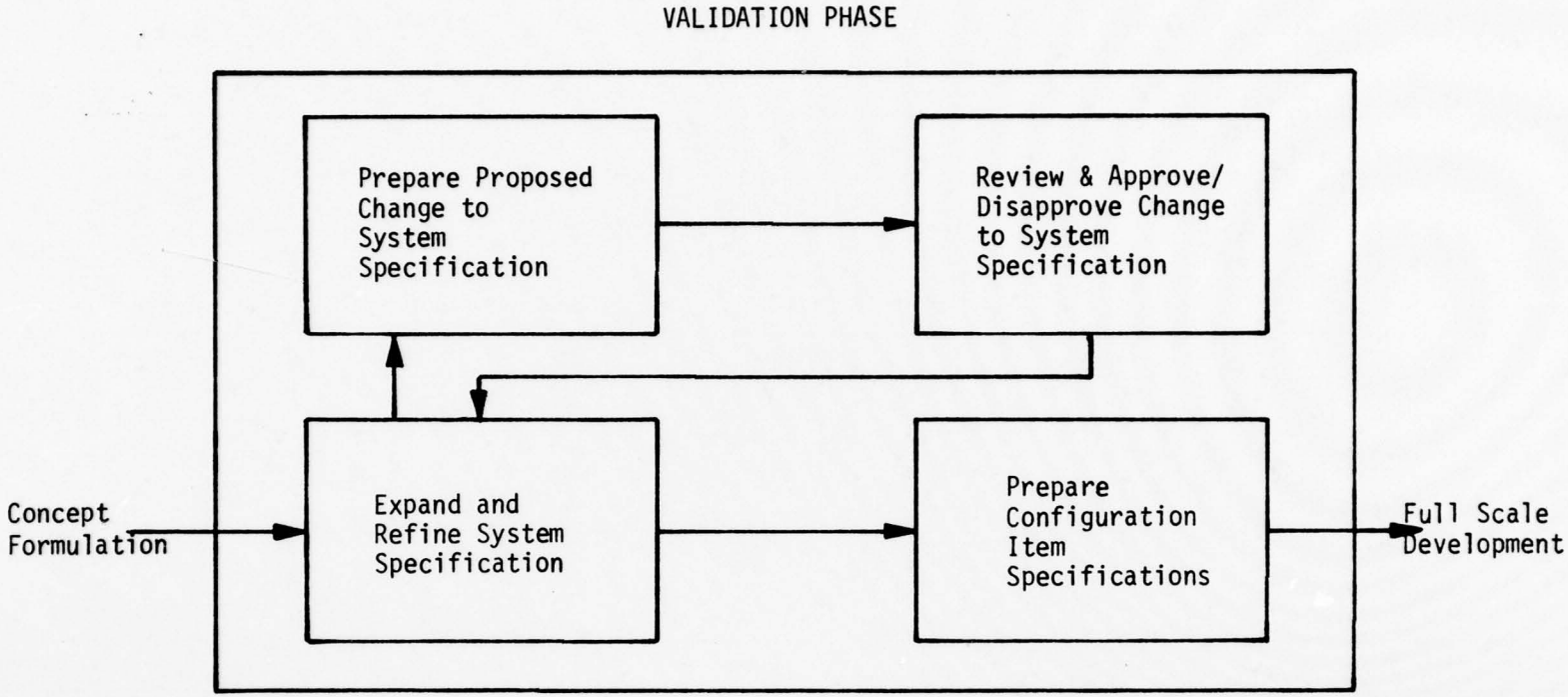


Figure 7. Validation Phase, Air Force Manual 375-7

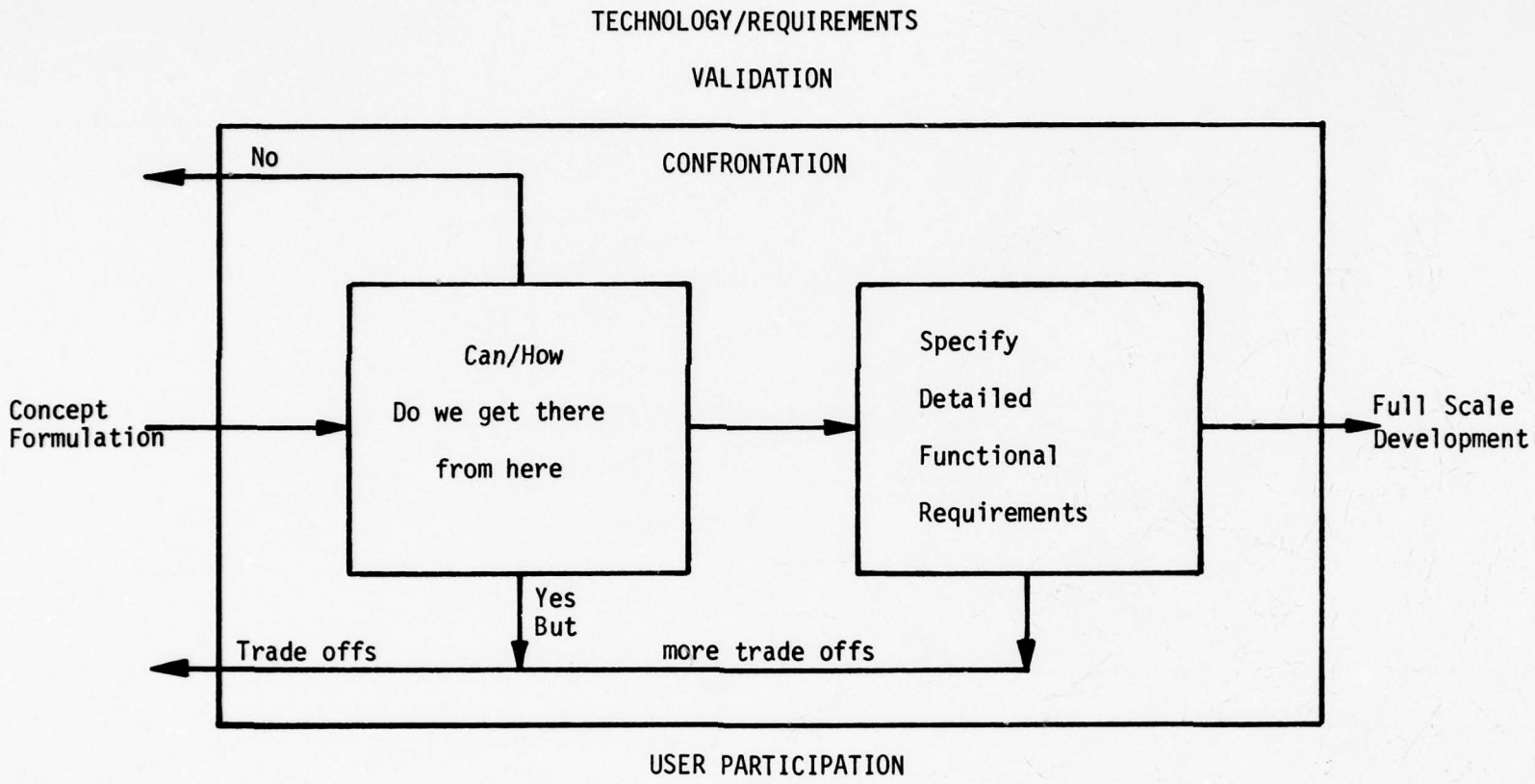


Figure 8. Rand Corporation Depiction of the Validation Phase

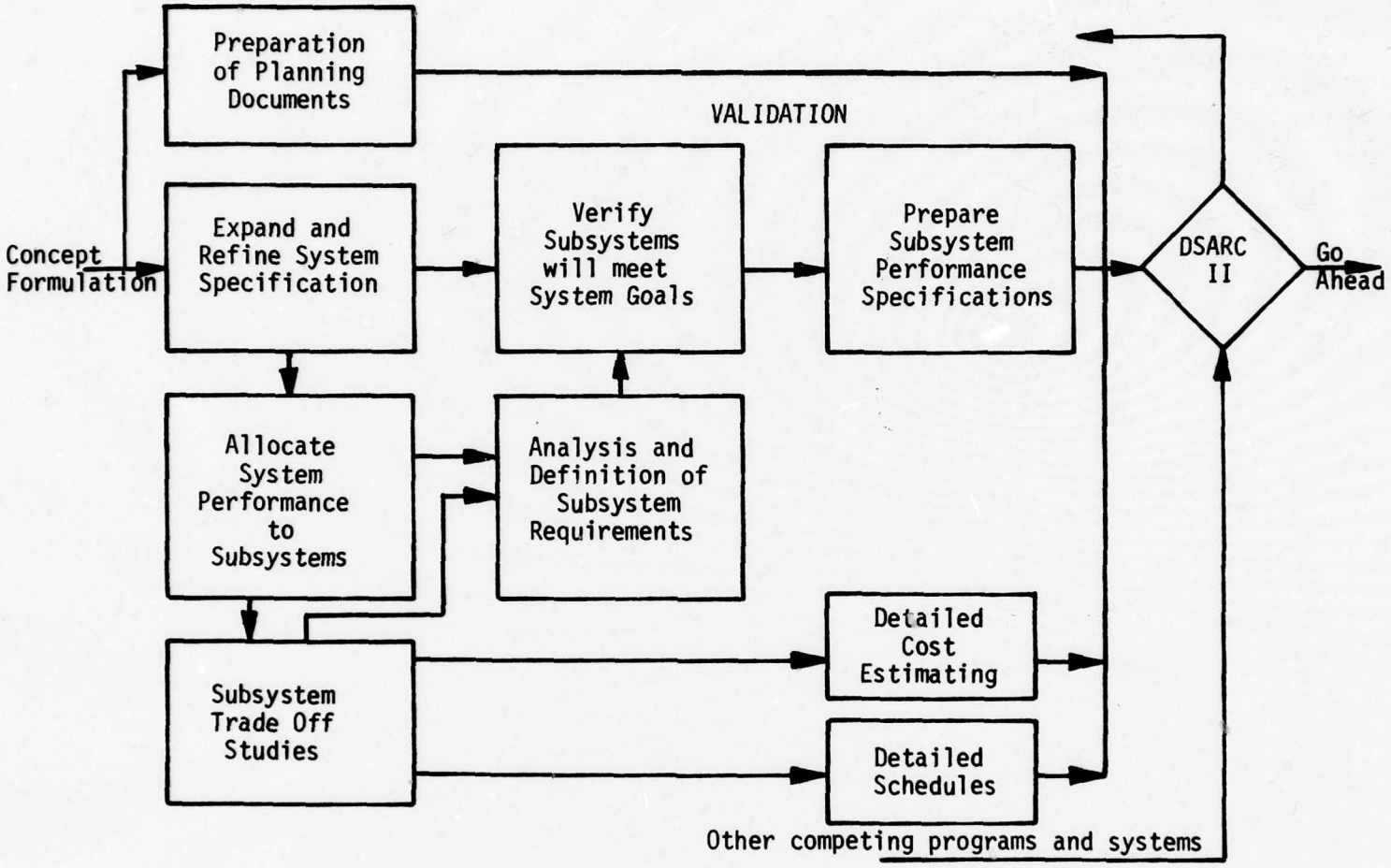


Figure 9. Detailed Validation Phase Activities

analysis through the Validation Phase. The Development Phase starts with coding, testing, debugging and parallel design efforts.

Although the above sequences may be desirable, instances of this sequence happening are very few and far between. The reader may be taken aback by this somewhat radical statement. I will not attempt to prove this statement, but simply cite the authorities and some of the evidence that points to this conclusion.

The author does not wish to say that some analysis of software does not occur during the Concept Formulation and Validation Phases; but compared to the indepth paper studies, analysis and even exploratory development of hardware, software studies are so minimal as to be almost nonexistent. Software is studied and analyzed only enough to establish two facts. First, the system can and should be controlled to some degree by software; second, approximate software sizing activities are accomplished to give a rough order of magnitude of the software costs.

A major function of the Validation Phase is to establish the performance and design requirements specification. It is a recognized fact that the programming of software is an abstract process. The logic of software comes from the creative imagination of the programmer's mind. No two people or groups of people think in the same thought patterns. I think that this could well lead us to conclude that the people who are going to do the actual programming are also the people who must do the design analysis necessary to establishing software design and performance requirements.

Boehm in his excellent and often quoted article referred to the software as secondary to hardware. His emphasis was that software decisions and requirements should not wait until after the critical hardware

decisions are made, but should be accomplished first. Boehm goes on to state that 35% of the total software costs are spent on software analysis. The cost of software is basically the cost of people, because software comes from the minds of men. Both Boehm and Brooks show that by adding men in an attempt to speed up a programming project, the project becomes more confused and actually takes longer. (1) (25) I think we are safe in stating that the cost of software in this sense can be related directly to the time required for those efforts. The assumption here is that the programming team size is constant. This lends credence to the assumption that 35% (analysis) of the total software effort certainly is not completely accomplished during the Validation Phase.

This author has personal knowledge of at least one project where the software development specification was not completed until at least two years after start of the Full Scale Development Phase.

A recent software study performed by the Johns Hopkins Applied Physics Laboratory found that,

> "Despite the implications in the DSARC II review that an adequate design and costing basis must exist, current directives are vague on the formal requirements for the validation phase of the acquisition process." (2,2-3)

The study went on to recommend that directives require comprehensive analysis and definition of software during the validation phase as well as hardware. For this recommendation to be made, obviously, software analysis and definition were not normally accomplished during validation, at least at the time of this study.

We also find that,

> "In the structure of the RDTE (Research Development Test and Evaluation) program, advanced development is stated to include all projects that have moved into development of hardware for test. This is generally in systems where basic research and

> exploratory development have been completed in all areas except software. Software exploratory development and basic tool building are usually done during the development stage." (6, 2-3)

The situation becomes even clearer if we look at some of the latest recommended changes. The Joint Office of the Secretary/Services Weapons System Steering Committee for the resolution of Software problems, recommended directives be established that require the following software studies be performed during the Concept Formulation and Validation Phases. (6)

1. Software Requirements
2. Risk Analysis
3. Planning
4. Preliminary Design
5. Interface Control
6. Integration

It was also recommended that the embedded computer software be mandatorially established as configuration items. Designation as a configuration item establishes the level of management concern, attention, control and tracking. Up until now the software could be designated as either a configuration item or a critical item. The term critical item designates a much lower level of concern, attention, control and tracking during the procurement process. A major difference between the two items is that configuration item development specifications are an input to Full Scale Development and critical item development specifications are a product of Full Scale Development. Another important recommended requirement was that an overall computer resource plan be developed before Full Scale Development. These three specific items are accomplished for the hardware portion of an Embedded Computer System prior to the

start of Full Scale Development. For software generally these items are a product of the Full Scale Development Process.

These facts lead us to the conclusion that the hardware and software activities during the Validation and the Development phases are significantly offset. We can define the activities of the Validation Phase for software as the analysis and definition activities (including exploratory development), basic tool building and the development of the software performance and design specifications, which occur during the Full Scale Development Phase. Figure 10 is a representation of the hardware/software offset in validation activities.

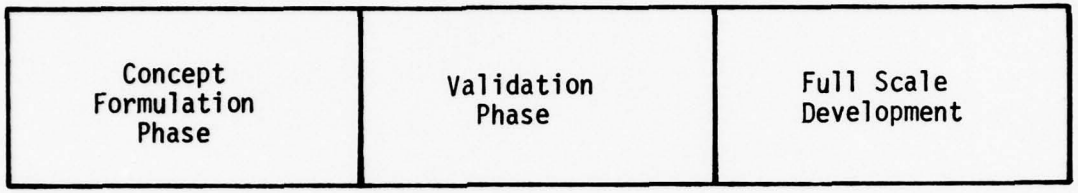
3. Full Scale Development. Because of the characteristics of the Validation Phase we must now break the Full Scale Development (FSD) Phase into four separate parts. In this phase of the acquisition cycle we will cover hardware development, software validation (analysis), software development and testing.

a. Hardware Development. The hardware phase of FSD covers all the work necessary to build one or more prototypes for testing. The development of hardware can be generally broken down into the stages listed below.

1. Preliminary Analysis and Design
2. Critical Analysis and Design
3. Building of Prototypes
4. Developmental Testing
5. Verification Testing
6. Validation Testing

During the preliminary design the developer and buyer gain an increased confidence in the feasibility, cost and performance of the

TRADITIONAL ACQUISITION PROCESS



EXPANDED DESCRIPTION OF ACQUISITION PROCESS

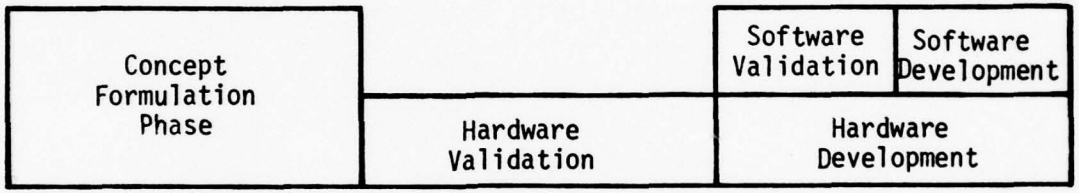


Figure 10. Partial Comparison of the Acquisition Process

system through an indepth look at the system concepts. This phase does not normally result in drawings for fabrication. Major modules and their interfaces are defined. Functional flow diagrams are prepared. General layout drawings may be prepared. Brassboarding and testing of critical components may occur. Maintainability and reliability requirements are addressed. Preliminary plans for further development, test, manufacture, installation, integration and support are developed. The results of the preliminary design review are inputs to the next stage, which is the critical design. (26)

In the critical design stage the recommendations of the preliminary design review are implemented. The work in this stage is the detailing of design and analysis necessary to build the prototype. Detailed drawings for fabrication and part selections are made. Basic packaging decisions are made. If the basic processor has not already been selected in the preliminary design phase, it will be selected here. Building of prototypes although listed separately, occurs during the critical design process. Developmental testing also starts during this phase. Often a number of preproduction prototypes will be fabricated to allow various activities to occur simultaneously. For example, because of the delay (or offset) in the beginning of software activities, the design, coding and implementation of the software may occur simultaneously with reliability, environmental and other developmental testing of the hardware.

The end of the critical design stage will be a critical design review. The detailed design is generally completed at this point. Because validation/verification and integration testing require the

integration of the software into the total system they will be covered after the software development.

b. Software Validation. The discussion thus far may have led the reader to believe that the software and hardware development processes are separate activities. The software analysis (validation) starts with the input of the overall performance requirements specification for the embedded computer system. An analysis is performed in conjunction with the preliminary design of the hardware to determine the software/hardware interactions required to supply the system performance functions. From these interactions the software operating and design concepts are completed and documented in a software performance and design specification. The primary reason why the initial software development should be termed validation is because the creation of the development specification is a validation function.

c. Software Development. The actual software development can be broken down into five stages.

1. Basic Tool Building
2. Preliminary Design
3. Critical Design
4. Code and Debugging
5. Developmental Testing

The basic tool building stage is used to build such support tools as compilers, environmental simulators, documentation aids, test case generators, test data, management systems, assemblers, system exercisers, standards enforcers, special computer consoles or other necessary tools. (27)

During the preliminary design stage, analysis and tradeoffs are performed to determine alternative approaches to the computer programming problem. The design approach is selected during this activity. The programming technique, such as top-down programming, ego-less programming or chief programmer team programming, is selected. Compatibility requirements are defined, such as interface definitions, timing, message formats and available computer memory. Other activities are:

1. Definition of Inputs/Outputs
2. Designation of Programming Tasks
    a. Components
    b. Modules
3. Data Base Description
4. Functional Flows are Created
5. Allocation of Storage
6. Costs and Schedules are Updated
7. Development of Initial Test Plans

At the end of the preliminary design a formal preliminary design review (PDR) is held prior to continuing. This may consist of briefings, discussions and documentation analysis to determine if the software development is ready to continue to the critical design stage.

During the critical design stage those detailed design activities are accomplished which are necessary prior to actual coding of the software. The individual and system program flows are finalized. Preliminary test plans and procedures are finalized. Preliminary test plans and procedures are submitted for approval. A major result of the critical (detail) design process is the compilation of the major portions of data necessary to describe the computer software product. This data

will become the proposed product specification at the end of the validation verification process.

The critical design review is now held to insure the design is sufficiently mature to permit start of the actual coding. Also the testing procedures and plans are a major focus of this review.

During the coding stage the flow charts are converted to lines of coded instructions. The programming process is usually accomplished on a module or subroutine basis. The module or subroutine is then desk checked for illegal expressions, logic errors and deviations from programming standards. The subroutines or modules are then put through developmental testing with the special software tools such as simulators or special test cases. After sufficient modules have been checked out they are then compiled and assembled into a larger computer program segment which can then be tested and the process is repeated until the entire software program has been compiled and assembled and developmental testing performed. The computer program now is ready to enter the formal testing stages. Thus far the principle outputs of the hardware and software development have been: (15)

1. Computer Program Development Specification
2. Test Plans and Procedures
3. Drawings
4. Flow Charts
5. Computer Input and Output Formats
6. Source Program Statement (listings)
7. Object Program in Machine Language

d. Testing. During the testing stage two types of tests are performed--verification testing and validation testing. Verification

testing is sometimes referred to as systems integration testing. During validation testing both the hardware and software are tested separately against their requirements as spelled out in the specifications. The object of this testing is to determine if the specified requirements will be met. After successfully completing the validation testing the software and hardware are combined together and subjected to verification testing. Verification testing is essentially the application of input/output analysis to results obtained under a simulated operational environment. The results are then used to determine the degree of satisfaction of the user's requirements. The testing methodology is generally as follows:

1. Conduct a sequence of tests.
2. Analysis of results to determine how well requirements are met.
3. Initiate modifications as necessary to correct deficiencies.
4. Continue/repeat tests as necessary until all test objectives are met.
5. Document the results.

The testing phase normally ends with an audit of all performance functions. This audit is used to document that the performance requirements have been met. Once the audit (Functional Configuration Audit) has been completed the system has been qualified for production. An audit is performed at the same time to establish the product identification. This audit (Physical Configuration Audit) is used to verify that the technical documentation is complete and a true description of the product qualified. The Figures 11 and 12 are taken from Air Force Manual 375-7 and the Rand Corporation study. We can see that neither figure gives us a clear understanding of the relationships between hardware and software

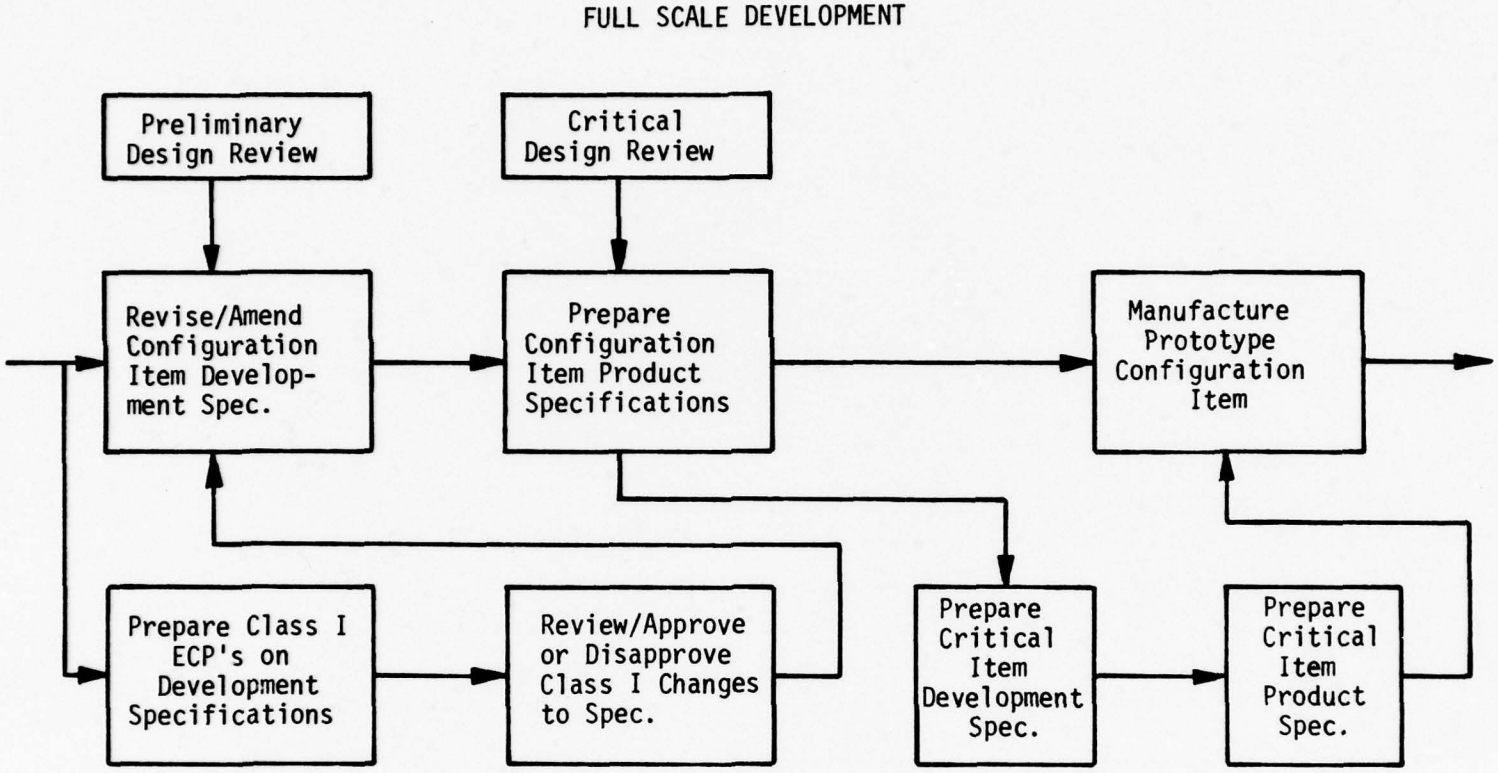


Figure 11. Depiction of the Design and Development Phase, Air Force Manual 375-7

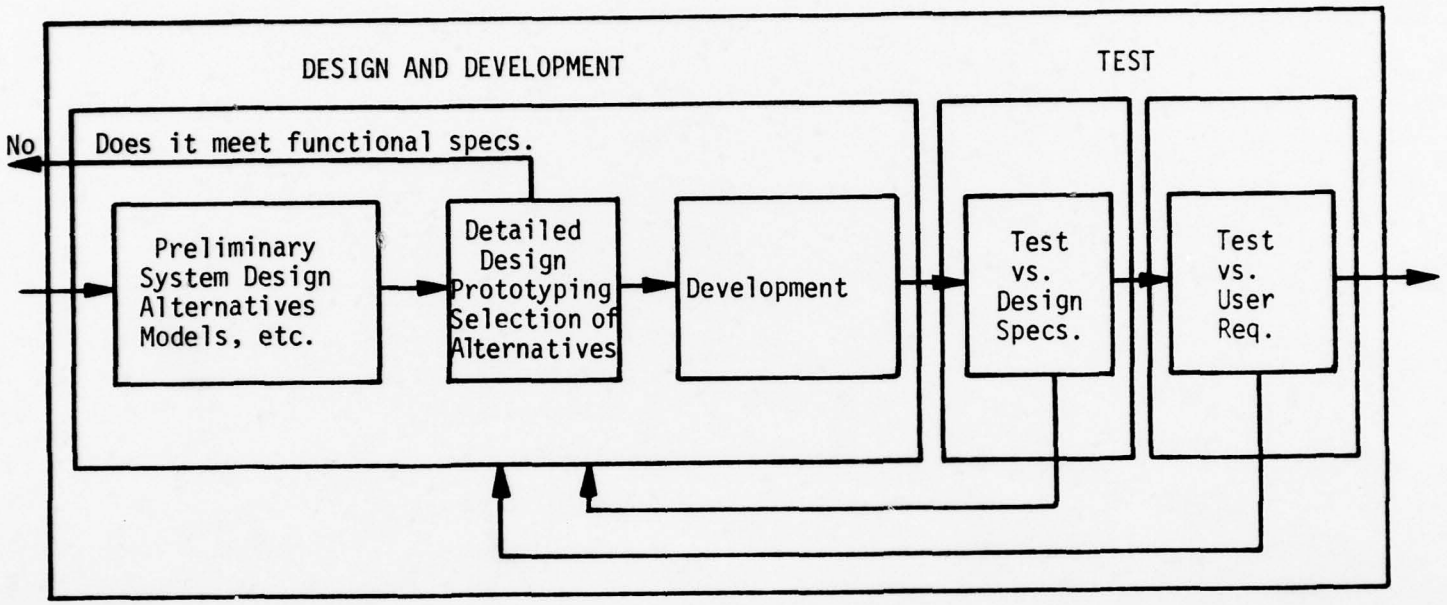


Figure 12. Rand Corporation Depiction of the Design and Development Phase

during the Full Scale Development phase of the procurement. Figure 13 shows the general relationships and the total life cycle covered to this point. Figure 14 shows the relationship among the hardware and software tasks. The dotted lines show relationships which although real are never formalized.

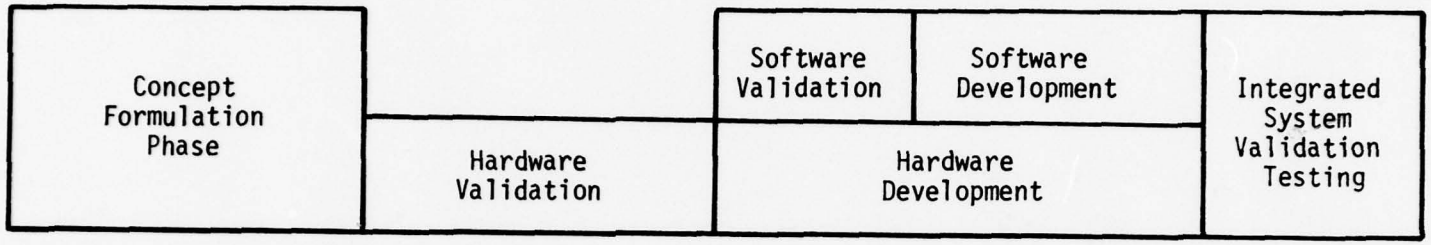
4. Production. In traditional procurement models, production is depicted as a block of time, after which operation and maintenance starts. Actually, in a multiple item procurement operation and maintenance will start very shortly after the first production item rolls off the assembly line. The first item or items are deployed to the field and run through a shakedown process to certify that they are operationally ready. So we can see from Figure 15 that there are parallel production, deployment and operational/maintenance phases. The relationship of these phases is described below.

During the production five separate and parallel activities are accomplished. The software is duplicated and accepted by the Air Force. The hardware items are sequentially manufactured over a long period of time. The hardware is sequentially accepted on a one-time basis. Activities necessary for the support of the hardware and software are accomplished. These production activities are depicted in Figures 16 and 17.

The software production activities amount to nothing more than duplication and acceptance of the software programs developed during the Full Scale Development phase. This duplication and acceptance can be accomplished in a matter of days.

During the software support stage two major activities can be expected to occur. First, the technical writing and editing is accomplished to finalize the user maintenance and operation documents. The

SYSTEM LIFE CYCLE



FULL SCALE DEVELOPMENT

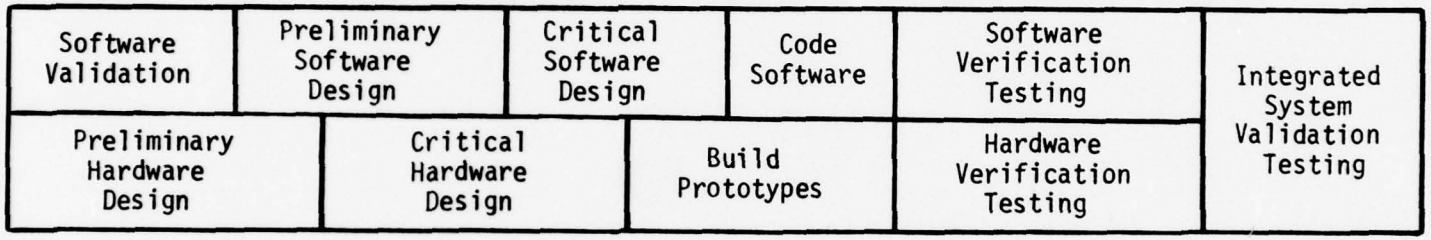


Figure 13. Task Relationships During Full Scale Development

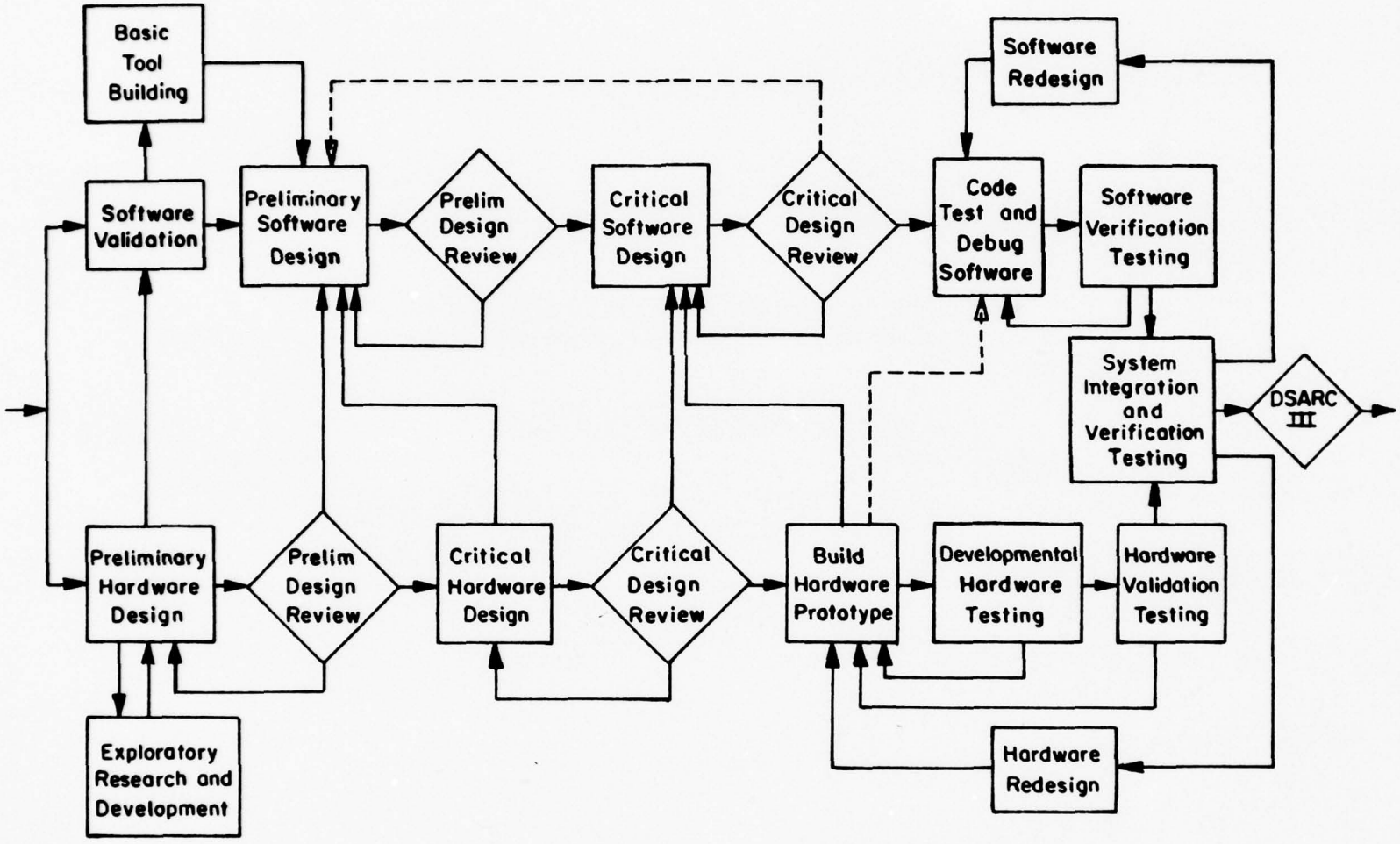


Figure 14. Detailed Hardware Software Development Activities

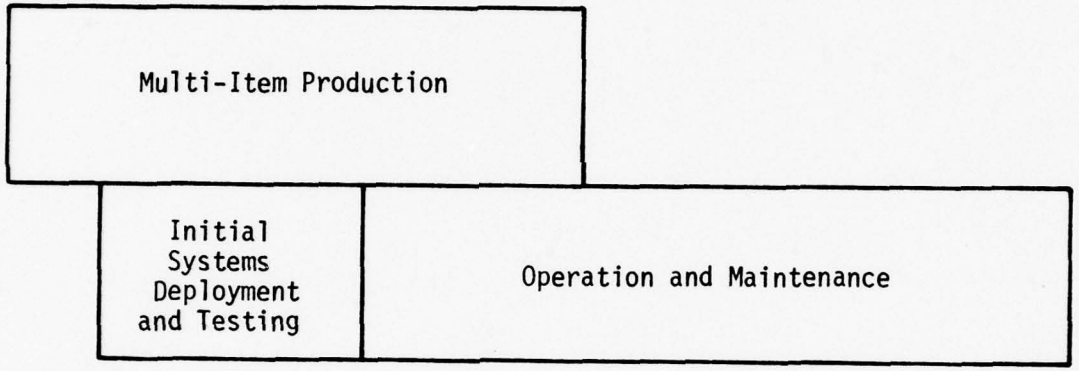


Figure 15. Production, Deployment, Operation and Maintenance Activities

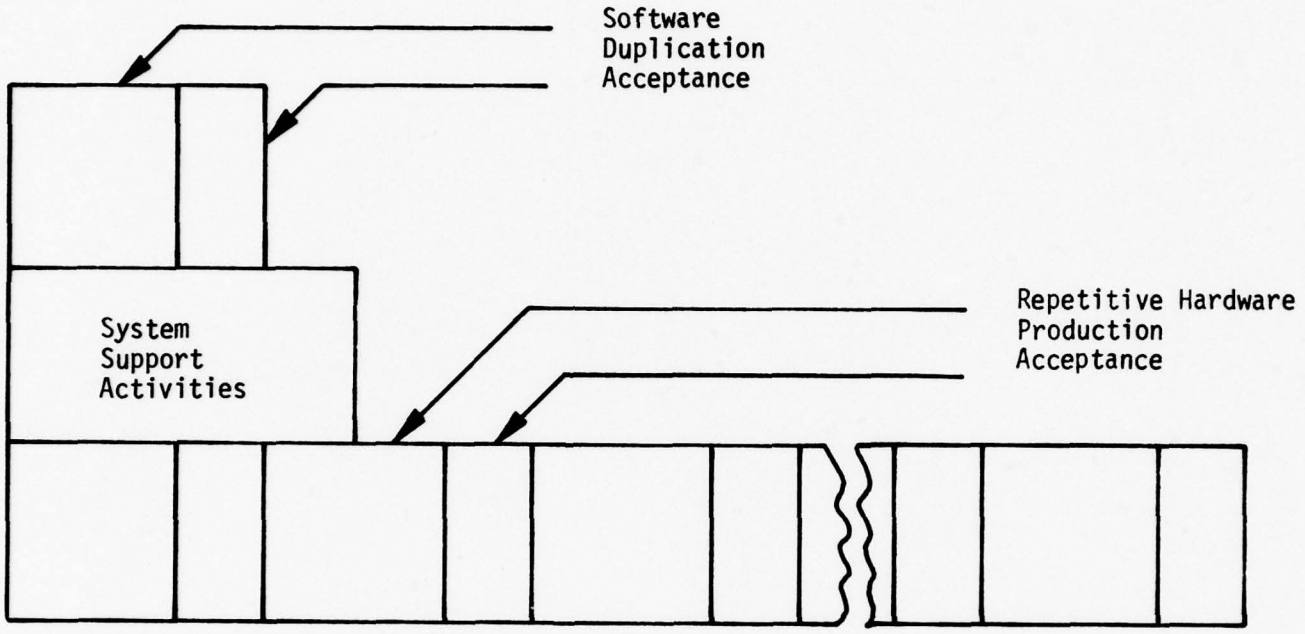


Figure 16. Production Phase Activities

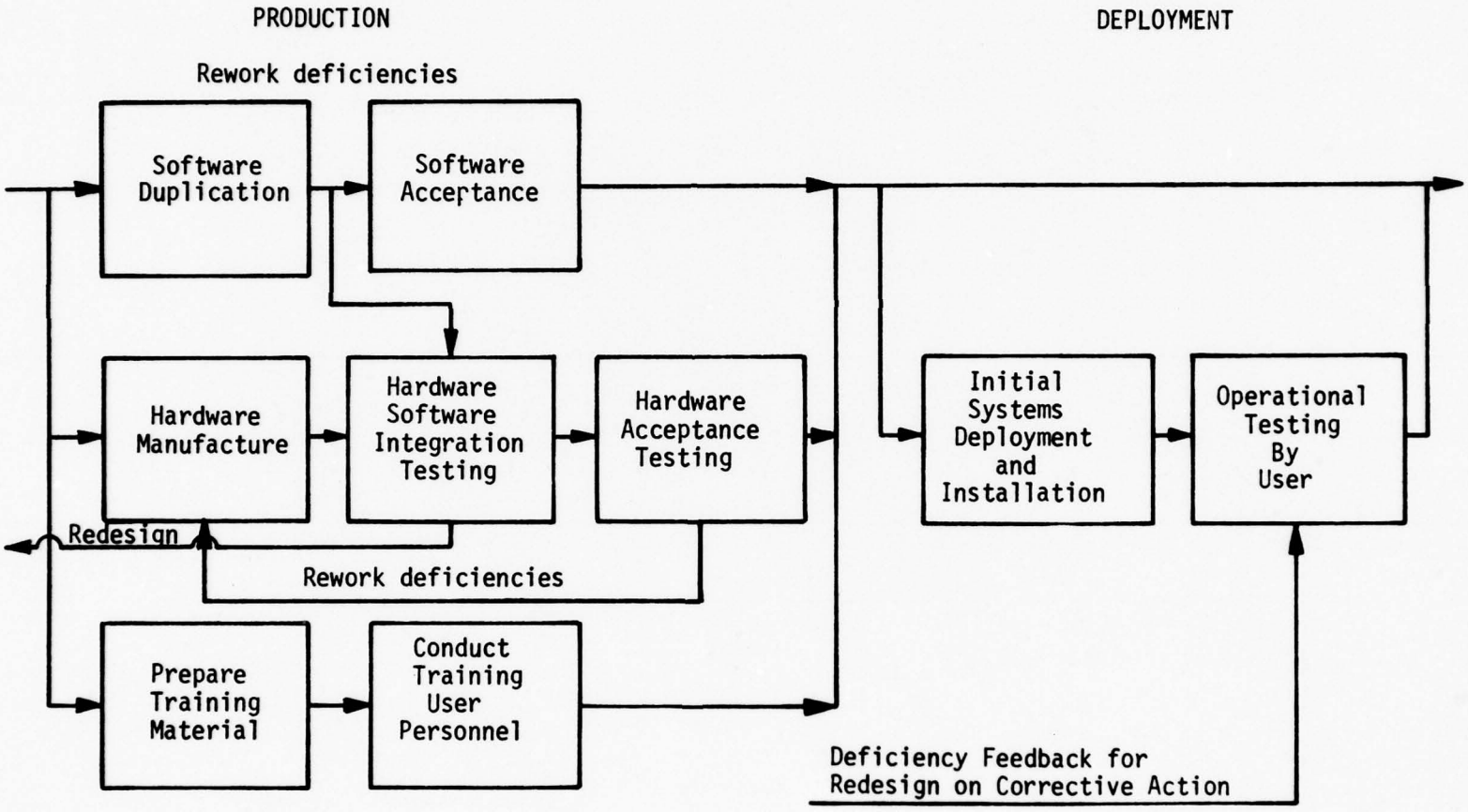


Figure 17. Detailed Production and Deployment Activities

second activity is that of training the user's personnel to operate and maintain the system. Training materials are prepared if they have not been previously developed during the Full Scale Development phase. The training program is then conducted. Training may be through briefing or formal classroom instructions. Two separate and distinct types of maintenance training must be provided for. The user's personnel must be trained to maintain the equipment and the software.

The last activity of the production phase is the acceptance of both the software and hardware. Previously during the development phase tests were designed and specified for acceptance. These tests are designed to demonstrate that the hardware and software comply with their production acceptance requirements. When the equipment and software pass their acceptance test we have established with confidence that they will perform satisfactorily in their operational environment.

5. Deployment. Deployment activities take place in two different ways. First, in the normal sense each individual system must be deployed to the field before normal operations can start. The other deployment concept is that of initial system deployment. The initial system deployment activities are essentially a shakedown cruise. The system is taken to the field and installed. In the case of aircraft systems, the equipment is normally installed at the factory. Field installation activity for aircraft systems is generally concerned with the placement, check out of support equipment and other preparations for support activities. After the systems and their support functions are in place, user testing is accomplished. The purpose of this testing is to assure that the system performs properly in the operational environment under live conditions. The output of this stage is a formal declaration by the user

that the system is ready for normal operations (Initial Operating Capability).

6. Operation, Maintenance and Retirement. Operation of an Embedded Computer System is the performance of its mission on a regular basis. Because operation is so readily understood we will skip discussion of operational activities. Because of the unique differences between hardware and software we must break the maintenance activities into two separate and distinct types of activities.

Hardware as a component will degrade over use and time. Mechanical parts wear out. Electrical components fail. Electrical characteristics of components may change requiring electrical adjustments and alignments. The hardware maintenance activities are to put the equipment back into the physical condition (specifications) that it was prior to the failure. This requires the use of technical data, testing equipment and a spare parts inventory.

Software does not fail in the same way that hardware fails. Software does not degrade over its use or time. It is not subject to the sudden and catastrophic failures that hardware is. One of the unique features of software is its consistency. Software will perform in the same manner every time. If the program has errors those errors will remain constant and always occur. In the operational phase the failures of software are not a cessation of the program's operation, but rather the recognition of errors which have been in the software all along. So we can say that the software maintenance activities are the correction of errors and modifications of the program to improve system capabilities. Modifications are also improvements which are necessary to upgrade systems capabilities, because of the changing operational environment.

Requirements are projected early in the system life cycle, but they must be continually revised as the situation changes. The revisions, corrections and modifications are in essence a redevelopment through the design, code and testing processes covered previously in the Full Scale Development Phase.

IV. MODEL EVALUATION

Research was conducted at the Air Force Systems Command Aeronautical Systems Division located at Wright Patterson Air Force Base, Ohio. The purpose to be served by this field research was to validate the various aspects of the thesis. The first topic to be investigated was the nature of the problems faced by the managers charged with the responsibility of acquiring ECS. Specifically, does ECS still represent a major acquisition problem? The second area to be investigated was the detailed knowledge of the participants. The knowledge level of the participants was used to gauge the usefulness of the answers received. In addition, the question to be answered was--can the expertise available at the working level be enhanced by a better understanding of the hardware/software interfaces during the acquisition process? And, the last subject covered was--does this attempt to model the ECS acquisition process provide an accurate and heretofore unavailable aid to managers' understanding of ECS?

The field research was carried out through the use of an interview questionnaire (Appendix A). The subjects for the interview were limited to only one organization; namely, the F-15 System Program Office. It is felt that because of the nature of the subject matter, a clearer picture could be formed by interviewing a select group from a single organization. In this way, continuity of thought was maintained throughout the interviews. Experienced practitioners from the various disciplines represented in a single Systems Program Office were chosen as participants. Appendix B provides a list of those who were interviewed.

There was a unanimous agreement among all the participants that the management of ECS acquisition represents a major problem within the DOD and industry. The specific problems given emphasis depended to a great extent upon what management function the interviewee was responsible for. For example, project managers had a great deal of concern for the question of how to measure the percentage completion of the software portion of the system. Engineering managers were concerned about how to validate and verify that the product is what it is supposed to be, while configuration managers were concerned about how to describe and document the software product such that the user would be able to organically modify and update the system after deployment. To this date, it was felt that none of the problems of Figure 1 had been completely resolved. Generally, the managers interviewed felt that their level of knowledge and expertise had increased significantly over the last few years. It was felt that while problems remained, a learning process was occurring and that things in a management sense are improving, even if only slowly.

All of the participants agreed that the present educational process for Air Force ECS managers is basically a learn-while-doing process, and that a basic understanding of the hardware/software interfaces must form the foundation for learning and improving the management of the ECS acquisition process. The managers were asked to evaluate Figure 18 as a representation of the concepts, tasks and hardware/software interrelationships during the ECS acquisition process. All of the participants felt that the model depicted in Figure 18 was essentially an accurate and adequate depiction of the ECS acquisition process in its present state. It is very interesting to note that the more directly involved with and more knowledgeable about ECS the participant was the more he

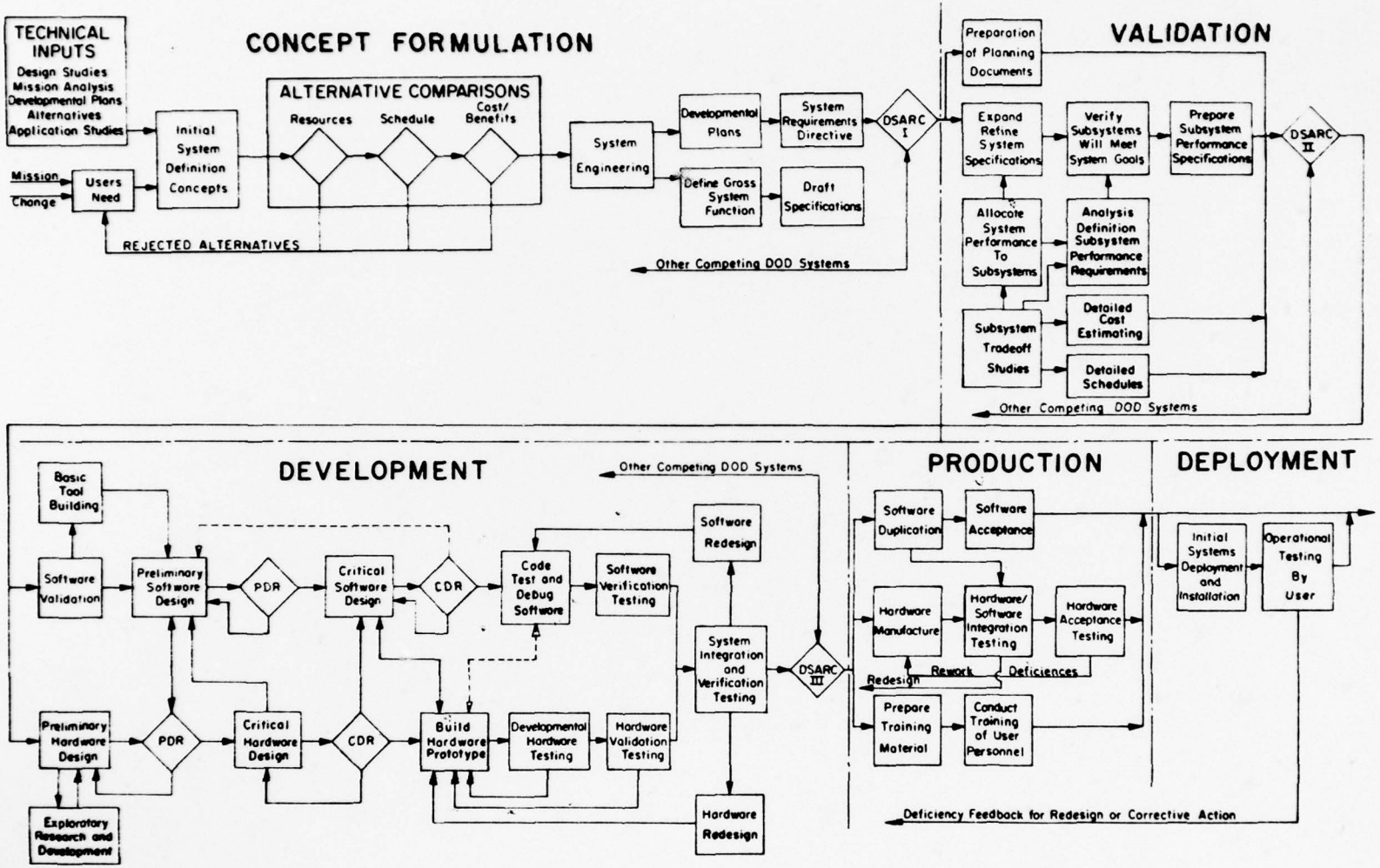


Figure 18. Detailed Activities of Embedded Computer Systems Acquisition

tended to agree that Figure 18 was an accurate description of the current acquisition model. One participant who had the most experience (fourteen years) was in complete agreement with the model. Other participants recommended minor changes. These changes were incorporated where they would add clarity, and not incorporated if they added confusing details.

V. CONCLUSION

As stated earlier, one of the major obstacles to effective management for ECS has been the lack of understanding of the complicated acquisition process. The transfer of knowledge and understanding from those who have some degree of expertise to those who do not has been almost nonexistent. This attempt to provide a basic reference document has been successful. The usefulness of this description of the ECS acquisition process, as it exists today, was verified by all of the participants during the field evaluation activities. The participants felt that the model could become a useful tool to ECS acquisition managers.

It was found during the field research that the indications of the software activities lagging behind the hardware activities were in fact true. It was verified that hardware decisions and activities were the driving force during most ECS acquisitions. The validation/verification testing activities were found to be confusing and in need of clarification. This need for clarification was also indicated by the profusion of different definitions found in the literature. Only the most knowledgeable managers were found to have an indepth appreciation for the software production process. Software production generally related well to hardware development activities. The duplication of software generally relates well to the reproduction of data. The thesis, although not advocating these definitions, was able to clearly show this relationship.

The research did not include a detailed analysis and description of the process for accomplishing the various tasks required during the

acquisition process. Further research should be performed to explain and describe in great detail how to perform each step during the acquisition process.

In the past, attempts to describe and explain acquisition tasks have been overly dependent upon emphasizing existing rules, regulations, requirements and documents to the detriment of understanding. I believe that the ECS acquisition tasks must be explained in the general terms of goals to be accomplished and how to arrive at those goals in a technical sense rather than just meeting administrative requirements. Each organization has differing administrative standards, and explaining tasks in terms of one organization's requirements is not readily transferable to another organization. A good example of the type of research needed might be Physical Configuration Audits (PCA) of ECS software. Generally speaking, the goal of a PCA is to establish that the documents describing an item are a true and accurate description. At present there is a great confusion as to what PCA means and how to accomplish it. Is PCA simply the measurement of the size and placement of holes on a (paper) computer tape? Or, is it a detailed analysis of flow charts and listings in comparison with an actual computer program to determine compatibility and accuracy?

Figure 1 breaks ECS costs down into hardware and software cost categories. Additional research is needed to validate these cost categories. Possibly ECS costs should be broken down into three separate categories--hardware, software and system costs. Hardware costs would be only those costs to develop and produce the hardware as a single entity. Software costs would be those costs to develop and duplicate the software as a single entity. System costs would be all of those

costs occurring because the hardware and software are wedded together as an integrated functioning system. In other words, those costs which would not have occurred if software and hardware were not integrated into a single system. To accomplish this task, an accounting system which will separate the costs must be developed (implemented). At present, few ECS programs have an accounting system which shows software costs.

It should be noted here that although the field evaluation showed that this model represents the current state of the acquisition process, all of the participants agreed with the author's assessment that this model is not adequate for today's needs. It is felt that the software must be given more attention, software efforts must start earlier, and a unified systems approach must be used for Embedded Computer Systems.

BIBLIOGRAPHY

1. Boehm, Barry W. "Software and its Impact: A Quantitative Assessment," Datamation, XIX, 5 (May 1973).

2. Kossiakoff, A. and Sleight, T.P., and others. "DOD Weapon Systems Software Management," Johns Hopkins University Applied Physics Laboratory Technical Report, APL/JHU SR 75-3, (May 1975).

3. Gansler, Jacques S., Deputy Assistant Secretary of Defense. "Comment," Defense Management Journal, XI, 4, (October 1975).

4. Davis, Malcom R. "Thoughts on Establishing Guidelines and Rationale for Workshop Output," *Findings and Recommendations of the Joint* Logistics Commanders Software Reliability Work Group, Vol. II, HQ AFSC TR 75-05, (November 1975), p. 39-42.

5. Wolverton, R.W. and Schick, G.T. "Assessment of Software Reliability," TRD Report TR 75-05, (November 1972).

6. Government Report. Findings and Recommendations of the Joint Logistics Commanders Software Reliability Work Group, ed. Manly, John H. and Lipow, Myron, Headquarters Air Force Systems Command Andrews Air Force Base, D.C. 20334, HQ AFSC TR 75-05, (November 1975).

7. *Government Report.* DOD Weapons Systems Software Acquisition and Management Study, ed. Asch, A., and others, The Mitre Corporation, McLean, VA, MTR-6908, (May 1975).

8. Proposed DOD Directive Statements of Software Acquisition Management Policies, Practices and Procedures, Enclosure (1), Chief Naval Operations Memorandum, Ser. 03110408, (July 23, 1975).

9. Gansler, Jacques S., Deputy Assistant Secretary of Defense. "Remarks Before the Symposium on Computer Software," Memorandum, (April 10, 1976).

10. An Air Force Guide for Monitoring and Reporting Software, ed. Hagan, S.R., The Mitre Corporation, Bedford, Massachusetts 01730, MTR-3051, (September 1975).

11. Software Acquisition Management Guidebook: Regulations, Specifications and Standards, ed. Connolly, J.T., The Mitre Corporation, Bedford, Massachusetts 01730, MTR-3080, (October 1975).

12. An Air Force Guide to Software Documentation Requirements, ed. Schoeffel, W.L., The Mitre Corporation, Bedford, Massachusetts, MTR-3180, (June 1976).

13. Zabriskie, Ronald J. "Development of Weapon Systems Computer Programs: Guidelines for Controlling During FSD," Study Project Report presented to the Faculty of the Defense Systems Management School, Fort Belvoir, VA, (November 1975).

14. Etheredge, Boyd. "Computer Software Management from the Point of View of the System Manager," a Research Study presented to Faculty of the Air Command and Staff College, Maxwell Air Force Base, AL, (May 1974).

15. Nelson, E.A. "Management Handbook for the Estimation of Computer Programming Costs," Systems Development Corporation, Santa Monica, CA, (March 1976).

16. Merwin, Richard E. "Software Management Through Product Control," A paper presented to the Conference Proceedings on Managing the Development of Weapons Systems Software," held at the Air Command and Staff College, Maxwell Air Force Base, AL, (May 1976).

17. Capps, Larry R. "Software Management and the Testing of Weapons Systems that contain an Embedded Computer System," a Study Project Report presented to the Defense Systems Management School, Fort Belvoir, VA, (November 1975).

18. Mathis, N.S. and Willmorth, N.E. "Software Milestone Measurement Study," Systems Development Corporation, Santa Monica, CA, (November 1973).

19. Bucciarelli, Marco A. "Technical Performance Measurement for Computer Software Development Programs," a Study Project presented to the Faculty of the Defense Systems Management School, Fort Belvoir, VA, (May 1974).

20. Defense Standardization Manual 4120.3 - M, Standardization Policies, Procedures and Instructions, Office of the Assistant Secretary of Defense, Washington, D.C., (January 1972).

21. Draft Air Force Regulation 800-14, Management of Computer Resources in Systems, Department of the Air Force, Washington, D.C. (May 10, 1974).

22. Searle, L.V. Software Acquisition and Management, Coure B, Systems Development Corporation, Santa Monica, California, TM-5365/103/00, (October 1974).

23. Air Force Systems Command Manual/Air Force Logistics Command Manual 375-7, Systems Management Configuration Management for Munitions and Computer Programs, Department of the Air Force H Quarters, Air Force Systems Command, Andrews Air Force Base, Washington, D.C., (March 1971).

24. The Computer Resource Management Study: Executive Summary, ed. Drezner, Stephen M. and Shulman, Hyman. The Rand Corporation, Santa Monica, California, (September 1975).

25. Brooks, Jr., Fredric P. "The Mythical Man - Month," Datamation, Vol. 20, Number 1, (December 1974).

26. Babcock, Daniel L. "Systems and System Engineering," Office of Industrial Development Studies and Extension Division, University of Missouri-Rolla, University of Missouri, (1972).

27. Dunham, Dale C. "Navy Airborne Tactical Software Management: Review and Evaluation," Study Project Report presented to the Defense Systems Management School, Fort Belvoir, VA, (November 1975).

VITA

Jerry Keith Watson was born on August 2, 1943, in Ames, Iowa. He received his primary and secondary education in Des Moines, Iowa. Upon graduation from high school in June 1961, he entered the United States Air Force. He received his college education while on active duty from the University of Nebraska, Lincoln, Nebraska, Glendale Community College, Phoenix, Arizona, University of Mississippi, Biloxi, Mississippi, Jefferson Davis Junior College, Gulfport, Mississippi and the University of Missouri-Rolla in Rolla, Missouri. He received a Bachelor of Science degree in Engineering Management graduating Cum Laude from the University of Missouri-Rolla in December 1972. Upon graduation he was commissioned a Second Lieutenant in the United States Air Force. He subsequently worked as a configuration management officer for the Deputy for F-15/JEPO System Program Office at Wright-Patterson AFB, Ohio for three years. He has been enrolled in the Graduate School of the University of Missouri-Rolla since June 1976.

APPENDIX A

INTERVIEW GUIDE

1. Name _____.
2. Position _____.
3. Establishment of qualifications.
   a. Are you now or have you been in the past, involved with the acquisition of Embedded Computer Systems (ECS)?
   b. How many years have you been associated with the acquisition of computer systems?
4. Do you think the acquisition of Embedded Computer Systems presents problems to management in general, and your organization specifically?
5. Do you agree that the spiraling cost of ECS is a major acquisition problem?
6. Which of the following would you say is the most pressing, urgent and significant when considering ECS acquisitions?
   a. Contracting methods
   b. Management techniques
   c. The state of the technical art
   d. Others. Please list.
7. Do you agree that some of the problems in ECS acquisition are due to the rapid growth in the use of ECS, which has not been accompanied by appropriate growth management techniques and procedures?

8. When considering management techniques and methodologies, which of the following are problematic? Indicate major problems (MP), no problem (NP) or unknown (UK).

   _____ a. Tracking the system's progress.

   _____ b. Defining the hardware and software requirements.

   _____ c. Understanding the hardware and software relationships at each phase of the system's acquisition cycle.

   _____ d. Defining the software product.

   _____ e. Validating and verifying.

   _____ f. Defining and then implementing milestones for the ECS acquisition.

9. Do you agree that at present there is not a clear understanding at the working level, of the hardware/software ECS acquisition process?

10. Do you agree that if the problems of cost and management of ECS are to be solved that a good place to start with is a clear understanding of the ECS acquisition process?

11. Would you characterize the preparation and training for ECS acquisition managers as extensive, or would you characterize the preparation as a learn-while-doing process?

12. Would you characterize your evaluation of the attached model as:

    a. Complete agreement.

    b. Total disagreement.

    c. Need for improvement. Please list your suggestions for improvement.

    d. Other comments concerning the future directions for further research and problem solving.

13. Sound management practices for the acquisition of software are often available, but are not always followed. True or False? Please explain.
14. The process of defining software requirements does not normally include total life cycle considerations. True or False? Please comment.
15. Software indirect costs are often much greater than software direct costs. True or False. Please explain your answer.
16. Meaningful management information is often unavailable when needed, because of a lack of consistent practices for feedback of software management information. True or False? Comment.
17. What portion of the total acquisition efforts, conceptualization through operation and maintenance, does the software represent?
18. Software requirements, risk analysis, planning, preliminary design, interface definition occur during the Full Scale Development activities. True or False? Please comment.
19. Hardware development and construction is normally initiated so early that software is often forced to accept changes (because of hardware problems) without appropriate engineering and design. True or False? Please explain your answer.
20. Since software is uniquely different from hardware, the management schemes, techniques and procedures set up for Hardware will not work for software.
21. Many hardware inadequacies can be easily offset by simple software changes. True or False? Please explain your answer.

22. Acquisition of software can be treated as a production-like process, similar to the procurement of standard off-the-shelf items of hardware.
23. Support of software in operational systems is much the same as maintenance of hardware. True or False? Please comment.
24. Acquisition management for Embedded Computer Systems must adopt a total systems approach to acquisition. True or False? Please explain your answer.
25. The interrelationships between hardware and software activities during all phases of a system's life cycle is well understood. True or False? Please explain your answer.
26. It has been asserted that software as opposed to hardware lies on the critical path of most Embedded Computer Systems procurements. Do you believe it would be desirable to have the software analysis and design start earlier in the acquisition process than it does now?
27. Some authors insist that hardware decisions and design determine and limit software alternatives. Do you believe that Hardware/Software trade offs are made, or are the software design concepts driven by hardware decisions already made? Please explain.
28. The management techniques, policies and procedures are not the source of most problems; rather, the implementation and understanding of life cycle procurement relationships at the working level is the source of most software problems. How do you feel about this statement?

APPENDIX B

LIST OF PEOPLE INTERVIEWED

1. Beauvais, Maurice, Lt. Col., USAF
   Radar Warning Receiver Project Manager
   Directorate for Development and Operations
   Deputy for F-15/JEPO Systems Program Office
   Wright Patterson Air Force Base, Ohio

2. Calves, Clifton, Major, USAF
   Chief Software TEWS Division
   Directorate of Integrated Logistics Support
   Deputy for F-15/JEPO Systems Program Office
   Wright Patterson Air Force Base, Ohio

3. Elliott, Gilbert, Captain, USAF
   Configuration Management Officer (TEWS)
   Directorate of Configuration Management
   Deputy for F-15/JEPO Systems Program
   Wright Patterson Air Force Base, Ohio

4. Grosso, Frank, Captain, USAF
   Chief Specification Maintenance Branch
   Directorate of Configuration Management
   Deputy for F-15/JEPO Systems Program Office
   Wright Patterson Air Force Base, Ohio

5. Konomos, George

   Technical Director AFAL/TEWS

   Air Force Avionics Laboratory

   Wright Patterson Air Force Base, Ohio

6. Thompson, Kenneth, Lt. Col., USAF

   Chief Configuration Control Division

   Directorate of Configuration Management

   Deputy for F-15/JEPO Systems Program Office

   Wright Patterson Air Force Base, Ohio

7. Wilson, Arthur J., Lt. Col., USAF

   F-15 Support Equipment Project Manager

   Directorate for Development and Operations

   Deputy for F-15/JEPO Systems Program Office

   Wright Patterson Air Force Base, Ohio